ELSEVIER

Theoretical Computer Science 286 (2002) 247–292

Theoretical Computer Science

www.elsevier.com/locate/tcs

Normal forms for algebras of connections ☆

Roberto Bruni, Fabio Gadducci *, Ugo Montanari

Dipartimento di Informatica, Università di Pisa, Corsa Italia 40, I-56125-Pisa, Italy

Abstract

Recent years have seen a growing interest towards algebraic structures that are able to express formalisms different from the standard, tree-like presentation of terms. Many of these approaches reveal a specific interest towards the application to the 'distributed and concurrent systems' field, but an exhaustive comparison between them is sometimes difficult, because their presentations can be quite dissimilar. This work is a first step towards a unified view: Focusing on the primitive ingredients of distributed spaces (namely interfaces, links and basic modules), we introduce a general schema for describing a normal form presentation of many algebraic formalisms, and show that those normal forms can be thought of as arrows of suitable monoidal categories.

Keywords: Concurrent and distributed systems; Diagrammatic specification techniques; Graphical formalisms

1. Introduction

Since models of computation based on the notion of free and bound *names* are widespread, the notion of *name sharing* is essential for several applications ranging from logic programming, λ-calculus, functional programming and process algebra with restriction (or name hiding mechanisms) to mobile processes (where *local* names may be communicated to the external world, thus becoming *global* names). We can think of names as links to communication channels, or to objects, or to locations, or to remote shared resources, or also to some *causes* in the event history of the system. In general, names can be freely α-converted, because the main information they offer is *sharing*.

☆ Research partly supported by the EC TMR Network *GETGRATS* (General Theory of Graph Transformation Systems) and by Esprit Working Groups *APPLIGRAPH*, *CONFER2* and *COORDINA*, through the University of Pisa. Research partly carried out during the stay of the second author at the Division of Informatics of the University of Edinburgh, sponsored by the British EPSRC grant no. R29375.

* Corresponding author.

E-mail addresses: bruni@di.unipi.it (R. Bruni), gadducci@di.unipi.it (F. Gadducci), ugo@di.unipi.it (U. Montanari).

0304-3975/02/$ - see front matter
PII: S0304-3975(01)00318-8

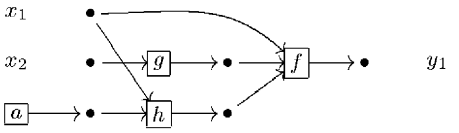


Fig. 1. Wire and box representation for the term $y_1 = f(x_1, g(x_2), h(x_1, a))$.

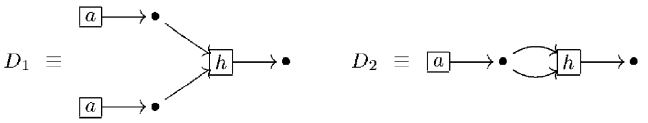


Fig. 2. Different diagrams for the term $h(a, a)$.

The simplest example is given by nonlinear contexts over a certain signature, where the same variable can occur twice or more: When the context is instantiated the actual values of such nonlinear parameters must be suitably duplicated so to fill all the 'holes' in the context.

An informal 'wire and box notation' may give an intuitive, visual understanding of the name sharing mechanism: Wires represent variables and the operators of the signature are denoted by boxes labeled with the corresponding operation symbols. Connection points between wires attached to boxes and variables are marked by the corresponding *sort* (i.e., the *type* of the wire). For instance, the diagram in Fig. 1 is the graphical representation of the term $y_1 = f(x_1, g(x_2), h(x_1, a))$ over the one-sorted signature

$$\Sigma = \{a : 0 \to 1,\ g : 1 \to 1,\ h : 2 \to 1,\ f : 3 \to 1\}$$

(containing the constant a, the unary operator g, the binary operator h and the ternary operator f) and variables x_1, x_2, where the unique sort is denoted by the symbol '●'.

Notice that wire duplication (e.g., of x_1) and wire swapping (e.g., of x_2 and a copy of x_1) are auxiliary, in the sense that they belong to any wire and box model, independently from the underlying signature. In general, the properties of the 'wire structure' (that is, of the overall connection topology) are far from trivial, and could lead to misleading system representations, whenever their interpretation is not well formalized. For example, let us consider the wire and box diagrams D_1 and D_2 in Fig. 2. In a *value oriented* interpretation, both D_1 and D_2 yield the same term $h(a, a)$. Instead, in a *reference oriented* interpretation, D_1 and D_2 define different situations: In the former the two arguments of the h operator are uncorrelated, while in the latter they point to the same *shared* location, whose content is 'a'.

Many mathematical structures have been recently proposed for expressing formalisms different from the ordinary tree-like presentation of terms. They range from the *flownomial calculus* of Ştefănescu [7, 39], to the *bicategories of processes* of Walters [22, 23], to the *pre-monoidal categories* of Power and Robinson [34], to the *action structures* of Milner [30], to the *interaction categories* of Abramsky [1], to the *sharing*

graphs of Hasegawa [20] and to the gs-*monoidal categories* of Corradini and Gadducci [10, 11], just to mention a few (see also [12, 16, 18, 35]). It is noteworthy that all these structures can be seen as enrichments of symmetric monoidal categories, which give the basis for the description of distributed environments in terms of wire and box diagrams.

We propose a schema for describing normal forms for this kind of structures, generalizing the one in [14] (and that bears some similarity to the *equational term graphs* of Ariola and Klop [2]), thus obtaining a universal framework where any informal diagram may find its unique standard representation. We describe distributed spaces as sets of *assignments* over sets of variables, distinguishing between four different kinds of assignment, each representing a basic functionality of the space, namely *input* and *output interfaces*, *basic modules*, and *connections* (also called *links*).

Interfaces are the only means through which a distributed space can communicate with the environment. Thus, distributed spaces can be seen as black boxes that can be composed in parallel (juxtaposing their interfaces) and sequentially (merging the output interface of a box with the input interface of a second box). Basic modules are a sort of primitive black boxes on which every other distributed space is built. Links can be thought of as cables that are used to allow basic modules to exchange information, i.e., links are some sort of communicating channels, describing the abstract topology of the system. As in the real world, where one has the necessity to distinguish between, e.g., one-to-one and one-to-many communications, the usage of links can be restricted in many ways. Changing the constraints on the admissible connections is the key to move between formalisms such as relations, partitions, partial orders, etc. For example, one can consider spaces where the information flow among basic modules is acyclic (the more general kind of such spaces is called *relational*), still retaining the full compositionality of the framework. A smaller class of distributed spaces is obtained by considering relational spaces equipped with one-to-one channels only, which is still closed under parallel and sequential composition. When links are regarded as transitively closed bidirectional channels, we get another relevant class of systems called *partition spaces*, whose links broadcast data back and forth among the interfaces they are connected to.

We call Σ-*spaces* the distributed spaces over a signature Σ whose operators define the basic modules provided by the system, and show that the classes of Σ-spaces we are interested in always have at least the structure of a symmetric monoidal category. We then establish a tight correspondence between Σ-spaces and the initial model of various categorical entities, implicitly recasting many formalisms proposed in the literature (usually in a set-theoretical way) for modeling distributed systems in a unified framework. Applications range from the embeddings of Petri nets and contextual nets, to term graphs, partial orders, relations, and partitions.

The structure of the paper is as follows: In Section 2 we give a short account of the various enrichments over monoidal categories (roughly corresponding to different formalisms for the semantics of distributed systems studied in the literature) that we want to embed in our concrete normal form representation. In Section 3 we formally

define Σ-spaces, equip them with two operations of parallel and sequential composition, and state some additional properties for a relevant class of acyclic Σ-spaces, called *relational*. In Section 4 we show how it is possible to have a normal form representation for most of the categorical models previously considered (hence, implicitly also for net processes, term graphs, open graphs, relations, labeled partial orders, etc.) in terms of acyclic spaces. In Section 5 we show that by dropping the acyclicity requirement it is possible to model both partitions and contextual nets in a suitable way. The relevant point is that all the classes under consideration are characterized by simple restrictions on the admissible links of Σ-spaces: Building on that, in Section 6, we draw some conclusion and sketch future research aimed at the *integration* of the structures considered in this paper, via an implementation of their normal form representation as Σ-spaces.

2. A categorical view for different formalisms

We recall here a few categorical definitions. They represent suitable enrichments of monoidal categories and, as it is argued by various authors (and surveyed in e.g. [11, 18]), these structures allow to recast the usual notion of term over a signature in a more general setting. Moreover, the progressive enrichment of a basic theory with different, additional constructors generates a great variety of different model classes, where the notions of relation, partial order, partition, and many others can be represented and compared.

Definition 2.1 (*Signatures*). A *many-sorted hyper-signature* Σ over a set S_Σ of sorts is a family $\{\Sigma_{\omega,\omega'}\}_{\omega,\omega' \in S_\Sigma^*}$ of sets of operators. If S_Σ is a singleton, we denote the hyper-signature Σ by the family $\{\Sigma_{n,m}\}_{n,m\in\mathbb{N}}$.

We usually omit the prefix 'hyper'. When it is clear from the context that a set S is the underlying set of sorts of a signature Σ, we drop the subscript $_{-\Sigma}$.

A many-sorted signature Σ over S can be viewed as a graph G_Σ, whose nodes (also *objects*) are strings on S, and whose edges (also *arrows*) are the operators of the signature (i.e., G_Σ contains an edge[1] $f:\omega\to\omega'$ if and only if $f\in\Sigma_{\omega,\omega'}$).

Notice that for a set S, the set S^* of its strings is the underlying set of a monoid, where string concatenation $\_\cdot\_$ yields the monoidal operator, and the empty string ε is the neutral element. Thus, a *signature morphism* F from Σ to Σ' is just a graph morphism whose object component is a monoid homomorphism, mapping each element of S into an element of S'.

A chain of structural enrichments enhances the expressiveness of different classes of models. The first enrichment is common to all the formalisms we will consider, and introduces the sequential and parallel composition operators, together with all the

[1] We use $f:a\to b$ to denote an edge f with source a and target b.

arrows necessary for arbitrary permutations of objects (corresponding to the swappings of wires in the wire and box presentation). Then, the models of the resulting *symmetric theory* of Σ are just suitable symmetric monoidal categories [25], also equipped with the Σ-structure, i.e., where the operators of Σ are interpreted as arrows of the category.

We recall that a *category* $\mathscr{C}$ can be seen as a graph together with a total map $id_{\_}: O \to A$ that associates an arrow $id_a: a \to a$ to each object $a \in O$, and a partial, associative operation for sequential composition of arrows $\_;\_: A \times A \to A$ that is defined if and only if the target of its first argument matches the source of its second argument. Moreover, if $f: a \to b$ and $g: b \to c$ are arrows of $\mathscr{C}$ then $f; g: a \to c$ is an arrow, and for all $f: a \to b$ we have $id_a; f = f = f; id_b$. Abusing the notation we will often denote the arrow id_a by the object name a itself.

The obvious notion of morphism between categories (mapping arrows into arrows and objects into objects) must preserve the categorical structure (source, target, identities and sequential composition) and is called *functor*.

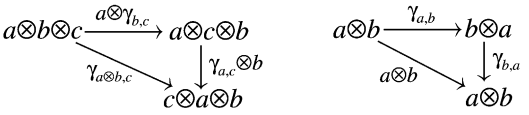
In what follows, we will sometimes refer to the *opposite* category $\mathscr{C}^{\mathrm{op}}$ of a category $\mathscr{C}$ as the category having the same objects as $\mathscr{C}$, but where the direction of the arrows is reversed (e.g., if $f: a \to b$ and $g: b \to c$ are arrows of $\mathscr{C}$, then $f^{\mathrm{op}}: b \to a$, $g^{\mathrm{op}}: c \to b$ and $g^{\mathrm{op}}; f^{\mathrm{op}} = (f; g)^{\mathrm{op}}$).

Definition 2.2 (*Monoidal and symmetric monoidal categories*). A (strict) *monoidal category* is a triple $\langle \mathscr{C}, \_ \otimes \_, e \rangle$, where $\mathscr{C}$ is the underlying category, the *tensor product* $\_ \otimes \_: \mathscr{C} \times \mathscr{C} \to \mathscr{C}$ is a functor satisfying the associative law $(t_1 \otimes t_2) \otimes t_3 = t_1 \otimes (t_2 \otimes t_3)$, and e is an object of $\mathscr{C}$ satisfying the identity law $t \otimes e = t = e \otimes t$, for all arrows $t, t_1, t_2, t_3 \in \mathscr{C}$.

A *symmetric monoidal category* is a 4-tuple $\langle \mathscr{C}, \_ \otimes \_, e, \gamma \rangle$, where $\langle \mathscr{C}, \_ \otimes \_, e \rangle$ is a monoidal category, and $\gamma: \_1 \otimes \_2 \Rightarrow \_2 \otimes \_1: \mathscr{C} \times \mathscr{C} \to \mathscr{C}$ is a natural transformation[2] satisfying the coherence axioms

$$\gamma_{a \otimes b, c} = (a \otimes \gamma_{b,c}); (\gamma_{a,c} \otimes b) \quad \text{and} \quad \gamma_{a,b}; \gamma_{b,a} = a \otimes b$$

for all objects a, b and c; *i.e.*, using a diagrammatic presentation



A functor $F: \mathscr{C} \to \mathscr{C}'$ between two (symmetric) monoidal categories is called *monoidal* if $F(t_1 \otimes t_2) = F(t_1) \otimes' F(t_2)$ and $F(e) = e'$; it is *symmetric* if in addition

[2] Given functors $F, G: \mathscr{A} \to \mathscr{B}$, a *transformation* $\tau: F \Rightarrow G: \mathscr{A} \to \mathscr{B}$ is a family of arrows of $\mathscr{B}$ indexed by objects of $\mathscr{A}$, $\tau = \{\tau_a: F(a) \to G(a) \mid a \in O_{\mathscr{A}}\}$. It is *natural* if for every arrow $f: a \to a'$ in $\mathscr{A}$, $\tau_a; G(f) = F(f); \tau_{a'}$ in $\mathscr{B}$.

Fig. 3. The symmetry $\gamma_{a,b}$ in the wire and box notation.

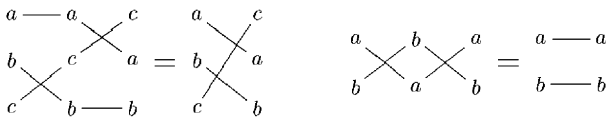


Fig. 4. Wire and box representation of the coherence axioms for symmetries.

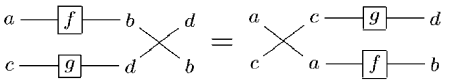


Fig. 5. Wire and box representation of the naturality axiom for symmetries.

$F(\gamma_{a,b}) = \gamma'_{F(a),F(b)}$. We denote by **SMCat** the category of symmetric monoidal categories (as objects) and symmetric functors (as arrows).

We emphasize that the axiom $\gamma_{a,b}; \gamma_{b,a} = a \otimes b$ makes γ a natural isomorphism whose inverse is the family of symmetries with reversed index arguments. Symmetries and the other auxiliary constructors we are going to present can be pictured in the wire and box notation as wire rearrangements. In our opinion, such representation illustrates very well the meaning of coherence axioms. In particular, the generic symmetry $\gamma_{a,b}$ is illustrated in Fig. 3, and the coherence axioms in Fig. 4 (the labels a, b and c just denote the typing information associated to the wires).

The naturality of γ is expressed by saying that for all arrows $f : a \to b$, $g : c \to d$ we have (see Fig. 5):

$$(f \otimes g); \gamma_{b,d} = \gamma_{a,c}; (g \otimes f).$$

Very informally, the meaning is that if one looks at several complex diagrams that essentially contain the same functionalities, then the most important thing one has to pay attention to is tracing the connection between the components, and listing the order of input and output nodes.

In the following, we will use just the diagrammatic presentation for the axioms that the auxiliary structure must satisfy, since they usually are more descriptive than their algebraic counterparts.

2.1. Enriching the monoidal structure

The constructive definition of algebraic theories [24] as symmetric monoidal categories, enriched with two natural transformations (in the terminology we adopt in the

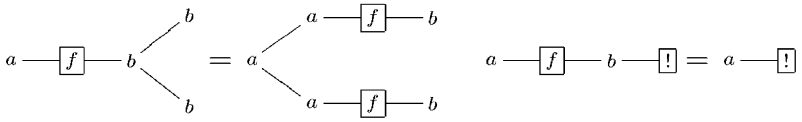


Fig. 6. The naturality axiom for duplicators and dischargers.

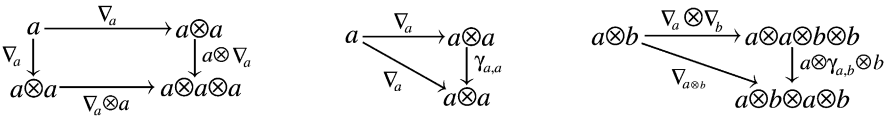
following, the *duplicator* and the *discharger*), dates back to the mid-1970s [21, 33], even if it has received a new stream of attention in these days. With respect to the usual presentation via universal properties (or equivalently, the ordinary description of terms by means of the meta-operation of substitution), it emphasizes and separates very nicely the 'link structure' from the Σ-structure, that is, from the textual occurrence in a term of the operators of the signature Σ. As shown in Fig. 6, the naturality axioms for these additional transformations express a controlled form of data sharing and garbaging. This intuition is further confirmed by the results in [11], where it is shown that, if these axioms are missing (i.e., if duplicators and dischargers are just *transformations*), then the corresponding *gs-monoidal* theory is the natural framework for the representation of *term graphs* rather than terms (the prefix 'gs' comes indeed from graph substitution).

In this section we propose a taxonomy for these enriched structures. We begin with the introduction of *share categories*.

Definition 2.3 (*Share categories*). A *share category* is a 5-tuple

$$\langle \mathscr{C}, \_ \otimes \_, e, \gamma, \nabla \rangle,$$

where $\langle \mathscr{C}, \_ \otimes \_, e, \gamma \rangle$ is a symmetric monoidal category and the transformation $\nabla : {}_{-1} \Rightarrow {}_{-1} \otimes {}_{-1} : \mathscr{C} \to \mathscr{C}$ is such that $\nabla_e = e$, and satisfies the coherence axioms expressed by the commutative diagrams below.



A *share functor* $F : \mathscr{C} \to \mathscr{C}'$ between two share categories is a symmetric functor such that $F(\nabla_a) = \nabla'_{F(a)}$. We denote by **ShCat** the category of share categories (as objects) and share functors (as arrows).

It is to be noticed that the naturality of the duplicator ∇_a (The operator is represented in Fig. 7 using the wire and box notation) is not required. The coherence axioms for duplicators in share categories are depicted in Fig. 8. The first two diagrams express a sort of associativity and commutativity for the duplicator, and the third diagram tells how the duplicator for a composed object $a \otimes b$ can be obtained by composing the duplicators for the subcomponents.

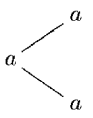


Fig. 7. The duplicator ∇_a in the wire and box notation.

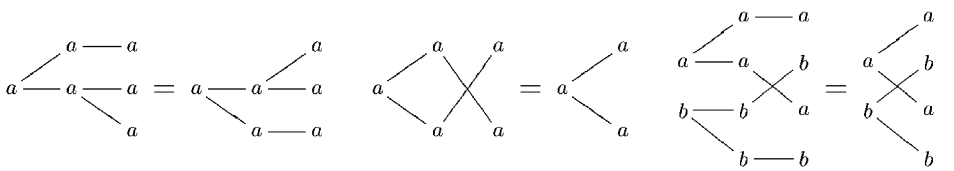


Fig. 8. The coherence axioms for duplicators in share categories.

Fig. 9. The discharger $!_a$ in the wire and box notation.

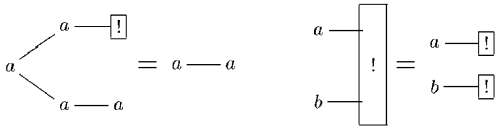
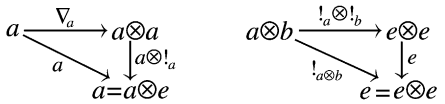


Fig. 10. The coherence axioms for dischargers in gs-monoidal categories.

Definition 2.4 (*GS-monoidal categories*). A *gs-monoidal category* is a 6-tuple

$$\langle \mathscr{C}, \_ \otimes \_, e, \gamma, \nabla, ! \rangle,$$

where $\langle \mathscr{C}, \_ \otimes \_, e, \gamma, \nabla \rangle$ is a share category, and $! : 1 \Rightarrow e : \mathscr{C} \to \mathscr{C}$ is a transformation such that $!_e = e$ and it satisfies the coherence axioms expressed by the diagrams below.



A *gs-monoidal functor* $F : \mathscr{C} \to \mathscr{C}'$ is a share functor such that $F(!_a) = !'_{F(a)}$. We denote by **GSCat** the category of *gs-monoidal* categories (as objects) and *gs-monoidal* functors (as arrows).

Dischargers and the associated axioms are illustrated in Figs. 9 and 10, respectively: The first diagram says that creating two links and then discharging one of them just yields the identity, the second diagram expresses the monoidality of !.

In the set-theoretical interpretation, the additional structures of share categories and of gs-monoidal categories correspond to the classes, respectively, of the converses of surjective (sort preserving) functions and of the converses of (sort preserving) functions, which provide initial models for the empty signature (cf. [8]). Whilst these statements will be made precise later, the use of the wire and box notation may give an intuitive understanding of this matter: When the object x in the source is connected to the object y in the target then y is in the counter-image of x. When only symmetries are considered (without duplicators and dischargers), the structure captures just bijections. Duplicators introduce non-injective correspondences, and dischargers can be used to abandon surjective matchings (seen from right to left).

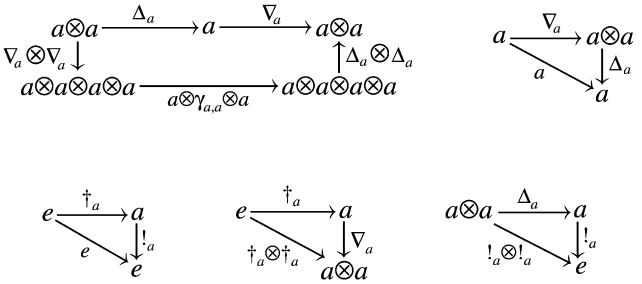
Interesting applications often require the presence of the categorical opposite of duplicators, which may be used to express a sort of *data matching*. Analogously, codischargers are introduced to represent the explicit *creation* of data. Several combinations are then possible, where only some of the operators are considered, and their mixed compositions are differently axiomatized, ranging from the *match-share categories* of [17] to the *dgs-monoidal categories* of [15, 16, 22]. We sketch now a survey of the topic, briefly commenting their role in the literature and the main differences between similar models.

We start with what we call an r-*monoidal category*: One of the various extensions, albeit with a different name, proposed in [8, 38, 39].

Definition 2.5 (*R-monoidal categories*). An r-*monoidal category* is an 8-tuple

$$\langle \mathscr{C}, \_ \otimes \_, e, \gamma, \nabla, !, \Delta, \dagger \rangle$$

such that $\langle \mathscr{C}, \_ \otimes \_, e, \gamma, \nabla, ! \rangle$ and $\langle \mathscr{C}^{\mathrm{op}}, \_ \otimes^{\mathrm{op}} \_, e, \gamma^{\mathrm{op}}, \Delta^{\mathrm{op}}, \dagger^{\mathrm{op}} \rangle$ are both gs-monoidal categories, satisfying the additional coherence axioms below.



An r-*monoidal functor* $F : \mathscr{C} \to \mathscr{C}'$ is a gs-monoidal functor (with respect to ∇ and $!$) such that also $F^{\mathrm{op}} : \mathscr{C}^{\mathrm{op}} \to \mathscr{C}'^{\mathrm{op}}$ is so (with respect to Δ^{op} and $\dagger^{\mathrm{op}}$). We denote by **RMCat** the category of r-monoidal categories (as objects) and r-monoidal functors (as arrows).

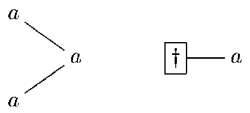


Fig. 11. The coduplicator Δ_a and the codischarger $\dagger_a$.

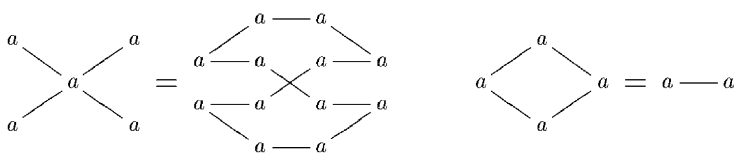


Fig. 12. The axioms $\Delta_a;\nabla_a=\nabla_{a\otimes a};(\Delta_a\otimes\Delta_a)$ and $\nabla_a;\Delta_a=a$.

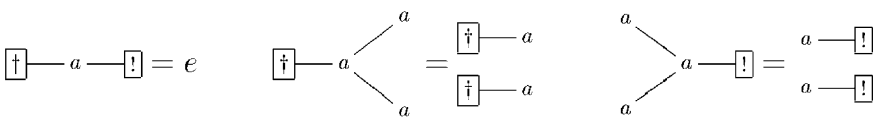


Fig. 13. The axioms $\dagger_a;!_a=e$, $\dagger_a;\nabla_a=\dagger_a\otimes\dagger_a$ and $\Delta_a;!_a=!_a\otimes!_a$.

The additional structure and axioms of r-monoidal categories, expressing the interplay between the gs and cogs structures, are illustrated in Figs. 11–13. For the sake of space, in the caption of Fig. 12 we have employed an equivalent but more compact—albeit less evocative—version of the axiom for r-monoidal categories $\Delta_a;\nabla_a=(\nabla_a\otimes\nabla_a);(a\otimes\gamma_{a,a}\otimes a);(\Delta_a\otimes\Delta_a)$, since by coherence in share categories we have $\nabla_{a\otimes a}=(\nabla_a\otimes\nabla_a);(a\otimes\gamma_{a,a}\otimes a)$.

The axioms we considered naturally embed the properties of relations, and the (partial) algebraic structure of r-monoidal categories yields a useful mathematical tool for their representation [8]. Again, the wire and box notation can provide some grasp on the intuition behind these representations. An object x in the source is related to an object y in the target if there is a wire between the two. Note that the number of wires (or paths) connecting two objects is irrelevant, since the axiom $\nabla_a;\Delta_a=a$ holds. Otherwise, one would consider multi-relations.

A stronger version of the axiom involving the composite $\Delta;\nabla$ is the basis for a different family of structures. In a certain sense, the stronger axiom establishes that duplicators and coduplicators embed a sort of transitive and symmetric closure of the relation, i.e., where only *reachability* is relevant in the link structure.

Definition 2.6 (*Match-share categories*). A *match-share category* is a 6-tuple

$$\langle\mathscr{C},\_\otimes\_,e,\gamma,\nabla,\Delta\rangle$$

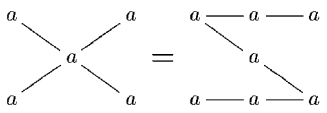
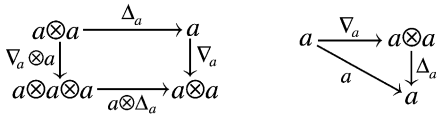


Fig. 14. The axiom $\Delta_a;\nabla_a=(\nabla_a\otimes a);(a\otimes\Delta_a)$.

such that $\langle\mathscr{C},\_\otimes\_,e,\gamma,\nabla\rangle$ and $\langle\mathscr{C}^{\mathrm{op}},\_\otimes^{\mathrm{op}}\_,e,\gamma^{\mathrm{op}},\Delta^{\mathrm{op}}\rangle$ are both share categories, satisfying the additional coherence axioms[3] below.



A *match-share functor* $F:\mathscr{C}\to\mathscr{C}'$ is a share functor (with respect to ∇) such that also $F^{\mathrm{op}}:\mathscr{C}^{\mathrm{op}}\to\mathscr{C}'^{\mathrm{op}}$ is so (with respect to Δ^{op}). We denote by **MShCat** the category of match-share categories (as objects) and match-share functors (as arrows).

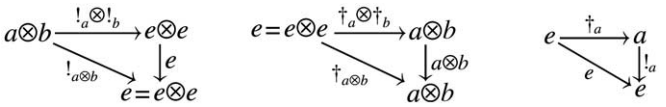
Match-share categories have been used in [17] to embed the algebraic properties of *processes* for *contextual nets* [31] and, more generally, they can be considered as models for *partition-based structures*. We talk about partitions because, in the set-theoretical interpretation, the axioms on the additional operators enforce a sort of transitive closure on the represented relation, thus defining an equivalence on the (disjoint) union of the source and target sets. This should be evident when looking at the wire and box representation in Fig. 14, which illustrates the more interesting axiom of match-share categories. Since dischargers and codischargers are not considered, in the initial model of the empty signature each equivalence class contains at least one object of the source and one of the target.

As in match-share categories we just introduce duplicators and coduplicators, we can do the symmetric choice so to define *new-bang categories*.

Definition 2.7 (*New-bang categories*). A *new-bang category* is a 6-tuple

$$\langle\mathscr{C},\_\otimes\_,e,\gamma,!,\dagger\rangle,$$

where $\langle\mathscr{C},\_\otimes\_,e,\gamma\rangle$ is a symmetric monoidal category, $!:1\Rightarrow e:\mathscr{C}\to\mathscr{C}$ is a transformation such that $!_e=e$, $\dagger:e\Rightarrow 1:\mathscr{C}\to\mathscr{C}$ is a transformation such that $\dagger_e=e$, and they satisfy the coherence axioms below.



[3] Using these axioms and the coherence axioms involving (co)duplicators and symmetries, also the equation $\Delta_a;\nabla_a=(a\otimes\nabla_a);(\Delta_a\otimes a)$ holds for any object a.

A *new-bang functor* $F:\mathscr{C}\to\mathscr{C}'$ is a symmetric monoidal functor such that also $F(!_a)=!'_{F(a)}$ and $F(\dagger_a)=\dagger'_{F(a)}$. We denote by **NBCat** the category of new-bang categories (as objects) and new-bang functors (as arrows).

To the best of our knowledge, new-bang categories have not been analyzed before. Basically, they provide the simplest enrichment able to deal with both *name creation* and *hiding*. As shown in Section 4.5, the analysis of this structure leads to an interesting representation of commutative monoids built on associative monoids (e.g., more concretely, multi-sets of strings over an alphabet, where the elements of the alphabet are regarded as unary operators of a signature).

Definition 2.8 (*P-monoidal categories*). A *p-monoidal category* is an 8-tuple

$$\langle\mathscr{C},\_\otimes\_,e,\gamma,\nabla,!,\Delta,\dagger\rangle$$

such that both $\langle\mathscr{C},\_\otimes\_,e,\gamma,\nabla,!\rangle$ and $\langle\mathscr{C}^{\mathrm{op}},\_\otimes^{\mathrm{op}}\_,e,\gamma^{\mathrm{op}},\Delta^{\mathrm{op}},\dagger^{\mathrm{op}}\rangle$ are gs-monoidal categories, $\langle\mathscr{C},\_\otimes\_,e,\gamma,\nabla,\Delta\rangle$ is a match-share category, and $\langle\mathscr{C},\_\otimes\_,e,\gamma,!,\dagger\rangle$ is a new-bang category.

A p-*monoidal functor* $F:\mathscr{C}\to\mathscr{C}'$ is a gs-monoidal functor (with respect to ∇ and $!$) such that also F^{op} is so (with respect to Δ^{op} and $\dagger^{\mathrm{op}}$). We denote by **PMCat** the category of p-monoidal categories (as objects) and p-monoidal functors (as arrows).

As for new-bang categories, we think that the set-theoretical aspects of p-monoidal categories have never been explicitly analyzed in the literature. Intuitively, a generic arrow from $a_1\otimes\cdots\otimes a_n$ to $b_1\otimes\cdots\otimes b_m$ (where the a_i's and the b_i's are 'basic' objects) represents some kind of partition of $\{a_1,\ldots,a_n,b_1,\ldots,b_m\}$, likewise in match-share categories. Notice however that here the initial model of the empty signature contains equivalence classes that do not include objects from both the source and the target (but empty equivalence classes are still avoided, since the axiom $\dagger_a;!_a=id_e$ holds). For example, the axiom $\Delta_a;!_a=!_a\otimes!_a$ of r-monoidal categories does not hold for partitions, because in the partition $\Delta_a;!_a$ both a sources belong to the same class, whereas in partition $!_a\otimes!_a$ they belong to disjoint classes.

Table 1 summarizes the structure of the various classes we have considered. In particular, the first four rows show whether (co)duplicators and (co)dischargers are required, while the remaining six rows illustrate the laws that regulate the interplay between duplicators, dischargers and their opposites. The columns are devoted to each one of the categories presented. Entries are marked with the symbol $*$ if the objects of the category in the corresponding columns (which are categories themselves) must satisfy the structural requirement associated to the row. The symbol — is used otherwise. Please note that, for the sake of readability, we presented the alternative version of the axiom for r-monoidal categories involving duplicators and coduplicators, namely $\Delta_a;\nabla_a=(\nabla_{a\otimes a});(\Delta_a\otimes\Delta_a)$, as already done in the caption of Fig. 12.

Table 1
A taxonomy of properties

	ShCat	GSCat	RMCat	MShCat	NBCat	PMCat
∇	*	*	*	*	—	*
Δ	—	—	*	*	—	*
!	—	*	*	—	*	*
†	—	—	*	—	*	*
$\nabla_a; \Delta_a = a$	—	—	*	*	—	*
$\Delta_a; \nabla_a = (\nabla_{a\otimes a}); (\Delta_a \otimes \Delta_a)$	—	—	*	*	—	*
$\Delta_a; \nabla_a = (\nabla_a \otimes a); (a \otimes \Delta_a)$	—	—	—	*	—	*
$\dagger_a; !_a = e$	—	—	*	—	*	*
$\dagger_a; \nabla_a = \dagger_a \otimes \dagger_a$	—	—	*	—	—	—
$\Delta_a; !_a = !_a \otimes !_a$	—	—	*	—	—	—

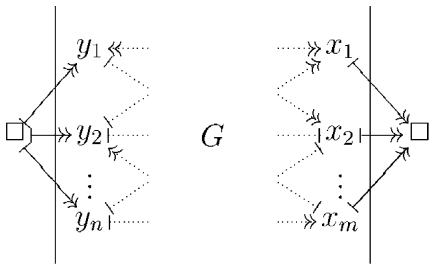


Fig. 15. Generic Σ-space.

3. The general model: Σ-spaces

We introduce now a concrete representation for the algebraic structures discussed in the previous section. It can be thought of as a normal form presentation of the less formal wire and box diagrams illustrated in the Introduction. Basically, we split the operative components of a diagram from the logical connectivity. The typical operative components of a distributed system are the *input* and *output interfaces*, and the *basic functional modules* (which can be viewed as n-to-m black boxes, giving the building blocks for the wire and box notation). The way these components can interact constitutes the logical part of the system. Suitable *link channels* can faithfully express this connectivity-related aspect. We write $z : s$ to say that the variable z has sort s.

Definition 3.1 (Σ-*Assignments*). Let Σ be a signature. A Σ-*assignment* over a bipartite set $\mathscr{Z} = \mathscr{X} \uplus \mathscr{Y}$ ($\mathscr{X}$ is called the set of *names*, and $\mathscr{Y}$ the set of *results*) of typed variables is any of the following *sentences*, where the x's, the y's and the z's range over $\mathscr{X}$, $\mathscr{Y}$ and $\mathscr{Z}$, respectively, and all the variables in a sentence are different.

Generator: $x_1, \ldots, x_n \overset{f}{\mapsto} y_1, \ldots, y_m$, where $x_i : s_i$ for $i = 1, \ldots, n$, $y_j : s'_j$ for $j = 1, \ldots, m$, and $f \in \Sigma_{\omega,\omega'}$ with $\omega = s_1, \ldots, s_n$, and $\omega' = s'_1, \ldots, s'_m$,

Link: $z_1 \mapsto\!\!\!\!\!\rightarrow z_2$, where $z_1 : s$ and $z_2 : s$,
Input: $\square \mapsto\!\!\!\!\!\rightarrow y_1, \ldots, y_m$, where $m \geqslant 0$,
Output: $x_1, \ldots, x_n \mapsto\!\!\!\!\!\rightarrow \square$, where $n \geqslant 0$.

We say that a variable z is *used* (respectively, *assigned*) if it appears in the left-hand (respectively right-hand) side of a sentence.

Definition 3.2 (Σ-*Spaces*). A *distributed space* over Σ is a set G of Σ-*assignments* such that

(1) G contains exactly one input sentence $\square \mapsto\!\!\!\!\!\rightarrow y_1, \ldots, y_m$, denoted by $in(G)$,
(2) G contains exactly one output sentence $x_1, \ldots, x_n \mapsto\!\!\!\!\!\rightarrow \square$, denoted by $out(G)$,
(3) all the variables in $in(G)$, $out(G)$ and $gen(G) = \{x_1, \ldots, x_n \overset{f}{\mapsto\!\!\!\!\!\rightarrow} y_1, \ldots, y_m \in G\}$ are different,
(4) all the variables in $link(G) = \{z_1 \mapsto\!\!\!\!\!\rightarrow z_2 \in G\}$ occur also in either $gen(G)$, $in(G)$ or $out(G)$,
(5) the set of links is closed transitively, in the sense that if $z_1 \mapsto\!\!\!\!\!\rightarrow z_2 \in G$ and $z_2 \mapsto\!\!\!\!\!\rightarrow z_3 \in G$ (with $z_1 \neq z_3$), then $z_1 \mapsto\!\!\!\!\!\rightarrow z_3 \in G$.

We call Σ-*spaces* the equivalence classes of distributed spaces over Σ up to injective renaming (since names in a distributed space are local, they can be safely renamed without affecting the interaction with the external environment).

We also let $\alpha_{\text{in}}(G)$ denote the tuple $\langle y_1, \ldots, y_m \rangle$ of variables occurring in the input sentence, as well as using $\alpha_{\text{out}}(G)$ for denoting the tuple $\langle x_1, \ldots, x_n \rangle$ of variables occurring in the output sentence.

Abusing the notation, a Σ-space is denoted by the same symbols of the distributed spaces in the equivalence class it represents. Given a distributed space G, its *flow relation* is the preorder induced over variables by the set of sentences; its *link relation* is the preorder induced over variables by the set of links.

A relevant class of Σ-spaces that enjoys some nice properties and that we shall exploit in the following is given by *relational* Σ-spaces.

Definition 3.3 (*Relational* Σ-*space*). A Σ-space G is called *relational* if

(1) the only kind of link sentences allowed consists of sentences of the form $y \mapsto\!\!\!\!\!\rightarrow x$ where y is a result and x is a name,
(2) the flow relation induced by the assignments in G is acyclic.

Note that, since links such as $y_1 \mapsto\!\!\!\!\!\rightarrow y_2$ and $x_1 \mapsto\!\!\!\!\!\rightarrow x_2$ are not allowed in relational Σ-spaces, then each nonempty chain of links must have length one.

Σ-spaces yield a monoidal category. Indeed, let $st : V^* \to S^*$ be the function mapping a list of typed variables into the corresponding list of types, e.g. $st(z_1, \ldots, z_n) = s_1, \ldots, s_n$ if $z_i : s_i$ for $i = 1, \ldots, n$. The objects are the elements of S^*, while each Σ-space G is viewed as an arrow $G : st(\alpha_{\text{in}}(G)) \to st(\alpha_{\text{out}}(G))$.

The parallel composition of two Σ-spaces G_1 and G_2 is always defined, yielding as a result the Σ-space $G_1 \otimes G_2$ that can be constructed as follows: Choose two distributed spaces in the classes of G_1 and G_2 such that their underlying sets of variables are disjoint (we can assume without loss of generality that G_1 and G_2 are already variable disjoint), then let

$$\alpha_{\text{in}}(G_1 \otimes G_2) = \alpha_{\text{in}}(G_1) \cdot \alpha_{\text{in}}(G_2),$$

$$\alpha_{\text{out}}(G_1 \otimes G_2) = \alpha_{\text{out}}(G_1) \cdot \alpha_{\text{out}}(G_2),$$

$$gen(G_1 \otimes G_2) = gen(G_1) \cup gen(G_2),$$

$$link(G_1 \otimes G_2) = link(G_1) \cup link(G_2),$$

where $\_ \cdot \_$ denotes string concatenation, and $\_ \cup \_$ denotes set union.

Proposition 3.1. *Let G_1, G_2 be two Σ-spaces. Then $G_1 \otimes G_2$ is a Σ-space.*

Proof. Properties (1) and (2) immediately follow by definition; property (3) follows from the fact that when merging the two spaces no name clash occurs; the last two properties (4) and (5) are ensured by the analogous properties on G_1 and G_2 and from the fact that all the variables in $link(G_1 \otimes G_2)$ appear either in $link(G_1)$ or in $link(G_2)$ (and similarly for *in*, *out* and *gen*). □

Proposition 3.2. *The parallel composition of Σ-spaces is associative.*

Proof. It trivially follows from the fact that inputs and outputs are represented as strings (juxtaposition is associative) and that Σ-spaces are sets of sentences (set union is associative). □

The empty Σ-space $G_\varepsilon = \{\Box \mapsto\!\!\!\!\to \varepsilon, \varepsilon \mapsto\!\!\!\!\to \Box\}$, where ε denotes the empty list of variables, is the unit element for parallel composition.

Proposition 3.3. *Let G be a Σ-space. Then $G \otimes G_\varepsilon = G = G_\varepsilon \otimes G$.*

The sequential composition of G_1 and G_2 is defined if and only if $st(\alpha_{\text{out}}(G_1)) = st(\alpha_{\text{in}}(G_2))$. For the sake of presentation, we assume G_1 and G_2 to be variable disjoint except for the variables in the output interface of G_1 and those in the input interface of G_2, i.e., $\alpha_{\text{out}}(G_1) = \alpha_{\text{in}}(G_2)$, and take the distributed space $G_1; G_2$ defined by

$$\alpha_{\text{in}}(G_1; G_2) = \alpha_{\text{in}}(G_1),$$

$$\alpha_{\text{out}}(G_1; G_2) = \alpha_{\text{out}}(G_2),$$

$$gen(G_1; G_2) = gen(G_1) \cup gen(G_2),$$

$$link(G_1; G_2) = merge(link(G_1), \alpha_{\text{out}}(G_1), link(G_2)),$$

where the composition of links $merge(L_1, W, L_2)$ is defined below (for W a vector of names and L_1, L_2 two sets of links, disjoint except for names in W).

First, we denote by $W[i]$ the ith variable in the list W, and write

$$z_1 \mapsto\!\!\!\!\rightarrow z_2 \mapsto\!\!\!\!\rightarrow z_3 \cdots z_n \mapsto\!\!\!\!\rightarrow z_{n+1} \in L,$$

when all the links $z_1 \mapsto\!\!\!\!\rightarrow z_2$, $z_2 \mapsto\!\!\!\!\rightarrow z_3, \ldots, z_n \mapsto\!\!\!\!\rightarrow z_{n+1}$ are contained in L. Then, we define $merge(L_1, W, L_2)$ as the set of links

$$\{z_1 \mapsto\!\!\!\!\rightarrow z_2 \mid z_1, z_2 \notin W, z_1 \mapsto\!\!\!\!\rightarrow W[i_1] \mapsto\!\!\!\!\rightarrow W[i_2] \cdots W[i_n] \mapsto\!\!\!\!\rightarrow z_2 \in L_1 \cup L_2\}.$$

Here $W[i_1] \mapsto\!\!\!\!\rightarrow W[i_2] \cdots W[i_{n-1}] \mapsto\!\!\!\!\rightarrow W[i_n]$ is a possibly empty sequence of links involving only variables in W, and we can safely assume that all the indices i_j are different. If the sequence is empty, then this means that either $z_1 \mapsto\!\!\!\!\rightarrow z_2 \in L_1$ or $z_1 \mapsto\!\!\!\!\rightarrow z_2 \in L_2$.

Intuitively, variables in $\alpha_{\text{out}}(G_1)$ (and $\alpha_{\text{in}}(G_2)$) are removed from the composition $G_1; G_2$, but their ingoing and outgoing links are propagated to the remaining variables, with respect to the matching $\alpha_{\text{out}}(G_1)[i] \leftrightarrow \alpha_{\text{in}}(G_2)[i]$, for $i = 1, \ldots, n$, of the variables in the 'merged' interfaces of G_1 and G_2. Note that this matching can introduce unexpected 'chains' of links, e.g., when $y \mapsto\!\!\!\!\rightarrow W[1]$, $W[1] \mapsto\!\!\!\!\rightarrow W[2] \in L_1$ and $W[2] \mapsto\!\!\!\!\rightarrow W[3]$, $W[3] \mapsto\!\!\!\!\rightarrow x \in L_2$ we have the link $y \mapsto\!\!\!\!\rightarrow x$ in the composed space. It is worth remarking that this definition preserves the transitive closure property (5) of Definition 3.2 on links.

The following technical lemmas will be useful for proving Proposition 3.7.

Lemma 3.4. *Let W be a list of names and let L_1 and L_2 be two sets of links, whose underlying sets of names are disjoint except for names in W. Then*
(1) $\{z_1 \mapsto\!\!\!\!\rightarrow z_2 \in L_1 \mid z_1, z_2 \notin W\} \subseteq merge(L_1, W, L_2)$; *and*
(2) $\{z_1 \mapsto\!\!\!\!\rightarrow z_2 \in L_2 \mid z_1, z_2 \notin W\} \subseteq merge(L_1, W, L_2)$.

Proof. Immediate, by definition of $merge(L_1, W, L_2)$ (considering empty chains of links between the names in W). □

Lemma 3.5. *Let W and W' be two lists of names, and let L_1, L_2 and L_3 be three sets of links such that names in L_1 and L_2 are disjoint except for names in W, names in L_2 and L_3 are disjoint except for names in W', and names in L_1 plus W are disjoint from names in L_3 plus W'. Then*

$$merge(L_1, W, merge(L_2, W', L_3)) = merge(merge(L_1, W, L_2), W', L_3).$$

The proof of Lemma 3.5 is simple but long and is postponed to Appendix A.1.

Since we mostly deal with relational Σ-spaces whose links have the form $y \mapsto\!\!\!\!\rightarrow x$ and therefore sequences such as $W[i_1] \mapsto\!\!\!\!\rightarrow W[i_2] \cdots W[i_{n-1}] \mapsto\!\!\!\!\rightarrow W[i_n]$ are either empty or contain just one variable $W[i]$ for a suitable index i, we characterize in this case

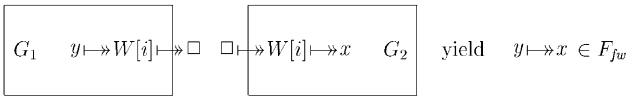


Fig. 16. Forward links for composition of relational Σ-spaces.

the set $link(G_1; G_2)$ as the union of three smaller sets

$$I_1 = \{y \mapsto\!\!\!\!\rightarrow x \in L_1\},$$

$$I_2 = \{y \mapsto\!\!\!\!\rightarrow x \in L_2\},$$

$$F_{fw} = \{y \mapsto\!\!\!\!\rightarrow x \mid y \mapsto\!\!\!\!\rightarrow W[i] \in L_1, W[i] \mapsto\!\!\!\!\rightarrow x \in L_2\},$$

with $x, y \notin W$. The sets I_1 and I_2 contain the links in L_1 and L_2 that do not involve variables in the merged interfaces (that disappear after the composition). The links in F_{fw} are obtained by matching the variables in the interfaces that are associated to the same index, as illustrated in Fig. 16.

Proposition 3.6. *Let G_1, G_2 be two Σ-spaces, such that $st(\alpha_{\text{out}}(G_1)) = st(\alpha_{\text{in}}(G_2))$. Then $G_1; G_2$ is a Σ-space.*

Proof. The property follows from the fact that variables in the merged interfaces can appear neither in $gen(G_1)$ nor in $gen(G_2)$. □

Proposition 3.7. *The sequential composition of Σ-spaces is an associative operation.*

Proof. By construction and by Lemma 3.5. □

Proposition 3.8. *Let G_1, G_2, G_1', G_2' be Σ-spaces such that G_i can be sequentially composed with G_i' for $i = 1, 2$. Then $(G_1; G_1') \otimes (G_2; G_2') = (G_1 \otimes G_2); (G_1' \otimes G_2')$.*

Proof. To prove this, it suffices to observe that the merged interfaces W_1 and W_2, between G_1 and G_1' and between G_2 and G_2', respectively, are disjoint. □

For each $\omega \in S^*$, the identity Σ-space G_ω is the unique Σ-space satisfying

$$st(\alpha_{\text{in}}(G_\omega)) = st(\alpha_{\text{out}}(G_\omega)) = \omega,$$

$$gen(G_\omega) = \emptyset,$$

$$link(G_\omega) = \{\alpha_{\text{in}}(G_\omega)[i] \mapsto\!\!\!\!\rightarrow \alpha_{\text{out}}(G_\omega)[i] \mid i = 1, \ldots, |\omega|\}$$

$$\cup \{\alpha_{\text{out}}(G_\omega)[i] \mapsto\!\!\!\!\rightarrow \alpha_{\text{in}}(G_\omega)[i] \mid i = 1, \ldots, |\omega|\}.$$

The proof that Σ-spaces G_ω indeed behave like identities with respect to sequential composition relies on property (5) of Definition 3.2. In fact, the composition introduces

some sort of transitive closure for the links relating variables in the merged interface, and for the identity to exist such closure cannot create new links.

Proposition 3.9. *Let G be a Σ-space. Then $G; G_{\mathrm{st}(\alpha_{\mathrm{out}}(G))} = G_{\mathrm{st}(\alpha_{\mathrm{in}}(G))}; G = G$.*

Since we are mainly interested in relational Σ-spaces, but any G_ω clearly violates both constraints for relational spaces, then other concrete spaces must play the role of identities for the subclass of relational Σ-spaces. By slightly abusing the notation, we denote again by G_ω the relational Σ-spaces behaving like identities with respect to sequential composition with other relational spaces, and we define them as follow

$$st(\alpha_{\mathrm{in}}(G_\omega)) = st(\alpha_{\mathrm{out}}(G_\omega)) = \omega,$$

$$gen(G_\omega) = \emptyset,$$

$$link(G_\omega) = \{\alpha_{\mathrm{in}}(G_\omega)[i] \mapsto\!\!\!\!\!\rightarrow \alpha_{\mathrm{out}}(G_\omega)[i] \mid i = 1, \ldots, |\omega|\}.$$

Proposition 3.10. *Let G be a relational Σ-space. Then $G; G_{\mathrm{st}(\alpha_{\mathrm{out}}(G))} = G_{\mathrm{st}(\alpha_{\mathrm{in}}(G))}; G = G$, with $G_{\mathrm{st}(\alpha_{\mathrm{out}}(G))}$ and $G_{\mathrm{st}(\alpha_{\mathrm{in}}(G))}$ the acyclic versions of identities.*

4. Σ-spaces for nets, term graphs and relations

In this section we show how many of the categorical models presented in Section 2 can be alternatively characterized as suitable acyclic Σ-spaces by imposing different requirements over the set of links. In particular, since these restrictions on the link structure are preserved by both parallel and sequential composition, those spaces can be considered as 'normal forms', that is, concrete representations of their categorical counterparts.

4.1. Symmetric monoidal

We begin by providing the class of Σ-spaces which characterize symmetric monoidal categories: Its elements must satisfy a tight requirement over links.

Definition 4.1 (*Symmetric Σ-space*). A Σ-space G is called *symmetric* if
(1) all the link sentences have the form $y \mapsto\!\!\!\!\!\rightarrow x$, for y a result and x a name;
(2) the flow relation induced by the assignments in G is acyclic;
(3) each variable is assigned exactly once in G;
(4) each variable is used exactly once in G.

Proposition 4.1. *Parallel and sequential composition of symmetric Σ-spaces yield symmetric Σ-spaces.*

Proof. For parallel composition the result is trivial. For sequential composition, let us consider G_1 and G_2 that can be concatenated. For each result y_1 in G_1 there is at most one link $y_1 \mapsto\!\!\!\!\rightarrow W[i]$ such that $W[i]$ is in the output interface of G_1. If such a link exists, then we take the *unique* link $W[i] \mapsto\!\!\!\!\rightarrow x_2$ in G_2 (that must exist for a variable x_2 not in the input interface of G_2). It follows that only the link $y_1 \mapsto\!\!\!\!\rightarrow x_2$ is created after the composition that involves y_1 and x_2, thus preserving the last two requirements of Definition 4.1. □

Note that the first two restrictions amount to requiring that G is a relational Σ-space. Moreover, the identities G_ω as defined for relational Σ-spaces are symmetric Σ-spaces, thus we get a monoidal category structure. To show that symmetric Σ-spaces define the arrows of a concrete symmetric monoidal category, we have to show what the symmetries are.

Definition 4.2 (*The Σ-space $G_{\omega,\omega'}$*). For each pair of strings $\omega, \omega' \in S^*$, with $|\omega| = n$, and $|\omega|' = m$, we define the symmetric Σ-space $G_{\omega,\omega'}$ as follows: Let Y and X be two lists of (pairwise disjoint) names such that $|Y| = |X| = n + m$, $Y[i] : (\omega \cdot \omega')[i]$, and $X[i] : (\omega' \cdot \omega)[i]$ for $i = 1, \ldots, n + m$, then

$$G_{\omega,\omega'} = \{\Box \mapsto\!\!\!\!\rightarrow Y,\ X \mapsto\!\!\!\!\rightarrow \Box\} \cup \{Y[i] \mapsto\!\!\!\!\rightarrow X[i + m] \mid i = 1, \ldots, n\}$$
$$\cup \{Y[i] \mapsto\!\!\!\!\rightarrow X[i - n] \mid i = n + 1, \ldots, n + m\}.$$

Proposition 4.2. *Symmetric Σ-spaces are the arrows of a concrete symmetric monoidal category, which is* (*isomorphic in* **SMCat** *to*) *the one freely generated by Σ.*

Let us make precise what we mean by 'freely generated' symmetric monoidal category. Algebraically, we refer to the initial term algebra over the binary operators $\{\otimes, ;\}$, with constants the names of the operators in Σ and the additional $\{\gamma_{a,b}\}$, for a, b sorts. Categorically, we refer to the image of Σ with respect to a suitable adjoint functor between the 'category of signatures' and **SMCat**, see e.g. [11] for a more exhaustive description. We present here just an informal sketch of the proof: We leave a detailed proof of the analogous result for gs-*monoidal spaces* (see Definition 4.3) to the appendix.

Proof (*Sketch*). Since symmetric Σ-spaces are acyclic, we already know (from Proposition 4.1) that they possess a monoidal category structure, with identities G_ω as defined on p. 18 (above Prop. 3.10). Hence we have to show that the family of Σ-spaces $\{G_{\omega,\omega'}\}_{\omega,\omega' \in S^*}$ is a natural isomorphism from $_{-1} \otimes _{-2}$ to $_{-2} \otimes _{-1}$, and satisfies the coherence axioms of Definition 2.2. This can be done directly by exploiting the definition of $G_{\omega,\omega'}$ and applying the definition of parallel and sequential composition. The freeness result relies on previous characterization results for symmetric monoidal categories as suitable (strongly concatenable) *Petri processes* [36, 37] and more recent analogous results for *pre-nets* [6], to which our spaces are equivalent.

First, let us give some terminology, to be used also in the sketched proofs of Propositions 4.5 and 4.8. For relational Σ-spaces we define the *depth* of a 'generator' sentence $x_1,\ldots,x_n \overset{f}{\mapsto\!\!\!\!\!\rightarrow} y_1,\ldots,y_m$ as the number of generator sentences in the longest flow relation chain that leads to any of the x_i's. The *sharing degree* of a variable is given by the number of outgoing links, plus one if the variable is an output, whilst the *matching degree* is given by the number of incoming links, plus one when the variable is an input. Thus, the proof can be summarized by four fundamental steps.

- We first define the functor H mapping the arrows of the symmetric category freely generated by Σ into our model by structural induction, i.e., since the basic arrows are identities, symmetries and generators, H is defined as follows:
  (1) each id_a is mapped to the identity space G_a;
  (2) each $\gamma_{a,b}$ is mapped to $G_{a,b}$;
  (3) each generator $f:\omega\to\omega'\in\Sigma$, with $|\omega|=n$ and $|\omega'|=m$ is mapped to

$$G_f=\{\Box\mapsto\!\!\!\!\!\rightarrow Y_\omega, Y_\omega[1]\mapsto\!\!\!\!\!\rightarrow X_\omega[1],\ldots,Y_\omega[n]\mapsto\!\!\!\!\!\rightarrow X_\omega[n],\ X_\omega \overset{f}{\mapsto\!\!\!\!\!\rightarrow} Y_{\omega'},$$

$$Y_{\omega'}[1]\mapsto\!\!\!\!\!\rightarrow X_{\omega'}[1],\ldots,Y_{\omega'}[m]\mapsto\!\!\!\!\!\rightarrow X_{\omega'}[m],\ X_{\omega'}\mapsto\!\!\!\!\!\rightarrow\Box\}$$

  for $X_\omega, Y_\omega, X_{\omega'}$ and $Y_{\omega'}$ disjoint tuples of variables (canonically chosen).
- We define a normal form for the arrows of the free symmetric monoidal category associated to Σ. The normal form consist of *arrows with maximal parallelism* that have the following shape:

$$s_1;(u_1\otimes t_1);s_2;(u_2\otimes t_2);\ldots;s_n;(u_n\otimes t_n);s_{n+1},$$

  where
  (1) the s_i's are permutations (i.e., arrows composed by identities and symmetries);
  (2) the u_i's are identities;
  (3) each t_i is the parallel composition of generators at depth i in the image through H of the arrow (respecting some predetermined, fixed total ordering of the operators in Σ).

  Notice that normal forms are unique only up to iso given by different choices of the permutations s_i;
- We show that Σ-spaces preserve reduction to normal form: Whenever two different normal forms are mapped into the same space, they can be proved equivalent. The proof works by induction on the length of the normal form, i.e., on the maximal depth of any generator;
- We conclude by defining the inverse functor of H, i.e., proving H surjective (thanks to the acyclicity of our structure, this can be easily done by considering any total sorting of the flow relation). □

The use of symmetric monoidal categories as an algebraic semantic framework for net processes [6, 29] guarantees that symmetric Σ-spaces offer a natural structure for those net-based systems where causality information plays a central role. Using

Σ-spaces, Petri net computations find an immediate representation in terms of places as sorts, tokens as variables, and transitions as operations. In fact, the link structure is needed for taking into account the dependency relation between tokens, and this intuition is further confirmed by the following result.

Proposition 4.3. *Let Σ be a many-sorted signature, and let $\omega, \omega' \in S^*$ be strings. The class of symmetric Σ-spaces with source ω and target ω', containing no generators, is in one-to-one correspondence with the class of* (*sort-preserving*) *bijective functions between the components of ω and those of ω'.*

We refer the reader to [13] for more details.

4.2. GS-monoidal

As illustrated in Section 2, gs-monoidal categories are symmetric monoidal categories enriched with suitable transformations for copying and discharging information, which lack the naturality axiom. In our setting, this enrichment reflects into a relaxation of the previous constraints over symmetric Σ-spaces.

Definition 4.3 (*GS-monoidal Σ-space*). A Σ-space G is called gs-*monoidal* if
(1) all the link sentences have the form $y \mapsto\!\!\!\!\rightarrow x$, for y a result and x a name;
(2) the flow relation induced by the assignments in G is acyclic;
(3) each variable is assigned exactly once in G.

The constraints are exactly the first three requirements stated in Definition 4.1. Notice that any symmetric Σ-space is also a gs-monoidal Σ-space.

Proposition 4.4. *Parallel and sequential composition of* gs-*monoidal Σ-spaces yield* gs-*monoidal Σ-spaces.*

The characterization of duplicators and dischargers is the intuitive one.

Definition 4.4 (*The Σ-spaces G^2_ω and G^0_ω*). For each string $\omega \in S^*$, with $|\omega| = n$, we define the gs-monoidal Σ-spaces G^2_ω and G^0_ω as follows: Let Y and X be lists of names such that $|Y| = n$, $|X| = 2n$, $Y[i] : \omega[i]$, $X[i] : \omega[i]$, and $X[i+n] : \omega[i]$, for $i = 1, \ldots, n$. Then

$$G^0_\omega = \{\Box \mapsto\!\!\!\!\rightarrow Y, \varepsilon \mapsto\!\!\!\!\rightarrow \Box\},$$

$$G^2_\omega = \{\Box \mapsto\!\!\!\!\rightarrow Y, X \mapsto\!\!\!\!\rightarrow \Box\} \cup \{Y[i] \mapsto\!\!\!\!\rightarrow X[i] \mid i = 1, \ldots, n\}$$

$$\cup \{Y[i] \mapsto\!\!\!\!\rightarrow X[i+n] \mid i = 1, \ldots, n\},$$

where we recall that ε denotes the empty list of variables.

We state now a correspondence result, confirming that gs-monoidal spaces provide a normal form for gs-monoidal categories. As for Proposition 4.2, we give here just a sketch of the proof; the full proof can be found in the appendix.

Proposition 4.5. *GS-Monoidal Σ-spaces are the arrows of a concrete gs-monoidal category, which is (isomorphic in* **GSCat** *to) the one freely generated by Σ.*

Proof (*Sketch*). As for the proof of Proposition 4.2, the acyclicity of gs-monoidal Σ-spaces together with Proposition 4.4 guarantees the monoidal category structure. Moreover, the Σ-spaces $G_{\omega,\omega'}$ defined in the previous section behave as symmetries also for gs-monoidal spaces (the proof is analogous to that sketched for Proposition 4.2). Therefore, we have just to show that the families of Σ-spaces $\{G^2_\omega\}_{\omega\in S^*}$ and $\{G^0_\omega\}_{\omega\in S^*}$ are transformations from the identity functor $_{-1}$ to $_{-1}\otimes_{-1}$, and to G_ε, respectively, verifying the coherence axioms of Definition 2.4. This can be done directly by exploiting the definition of G^2_ω and G^0_ω. As an important remark, it is trivial to verify that the naturality axioms are not satisfied by $\{G^2_\omega\}_{\omega\in S^*}$ and $\{G^0_\omega\}_{\omega\in S^*}$. To show the freeness of our model (see the appendix for a detailed proof) we rely on the results of [11], since gs-monoidal Σ-spaces offer a concrete structure corresponding to a normalized representation for *gs-graphs*. The proof technique is the standard one: First, a functor is defined by structural induction that goes from the freely generated gs-monoidal category to gs-monoidal Σ-spaces, with ∇_a and $!_a$ mapped into G^2_a and G^0_a, respectively; second, a normal form is defined for gs-monoidal terms and it is shown that whenever two different normal forms are mapped into the same Σ-space, then they can be proved equivalent; eventually, the inductively defined functor is proved surjective. Here we just remark that the normal form for the arrows of the free gs-monoidal category associated to Σ has the following shape:

$$g_1;(u_1\otimes t_1);g_2;(u_2\otimes t_2);\ldots;g_n;(u_n\otimes t_n);g_{n+1},$$

where:

(1) each g_i has the form $(\bigotimes_{j=1}^{n_i}\nabla_{a_{i,j}}^{k_{i,j}});s_i$, with s_i a permutation, $\nabla_a^0=!_a$, $\nabla_a^1=id_a$, $\nabla_a^2=\nabla_a$ and $\nabla_a^{k+1}=\nabla_a;(\nabla_a^k\otimes id_a)$ if $k\geqslant 2$ and where $k_{i,j}>0$ if $i\neq n+1$ (i.e., dischargers are allowed only in g_{n+1}); in particular each $k_{i,j}$ represents the sharing degree of the corresponding variable in the associated Σ-space, but when the sharing degree is 0, the discharger is postponed to the end of the expression representing the normal form;
(2) the u_i's are identities;
(3) each t_i is the parallel composition of generators at depth i in the Σ-space associated to the arrow (respecting some predetermined, fixed total ordering of the operators in Σ).

To some extent we take as normal forms those terms with *maximal parallelism and minimal duplication*. □

The main result of [11] states that the free gs-monoidal category over a (one-sorted, ordinary) signature is isomorphic to the class of (ranked) *term graphs* labeled over it.[4] It is in this setting, as well as in the presentation of *open graphs* [32], that we recover the intuitive interpretation of copying and discarding as suitable operations over graph-based structures. In general terms, the enrichment allows for a more complex link structure, analogous to one-to-many broadcasting, as suggested by the following result.

Proposition 4.6. *Let Σ be a many-sorted signature, and let $\omega, \omega' \in S^*$ be strings. The class of* gs-*monoidal Σ-spaces with source ω and target ω', containing no generators, is in one-to-one correspondence with the class of* (*sort-preserving*) *functions between the components of ω' and those of ω.*

We refer the reader to [11] for more details.

4.3. R-monoidal

Due to the presence of both coduplicators and codischargers in the relational model, we have no restriction on the number of incoming and outgoing links in the corresponding version of Σ-spaces. The weaker constraint considered here just involves the global structure of the link sentences: We recall now the definition of relational Σ-spaces (see Definition 3.3).

Definition 4.5 (*Relational Σ-space*). A Σ-space G is called *relational* if
(1) all the link sentences have the form $y \mapsto\!\!\!\!\rightarrow x$, for y a result and x a name;
(2) the flow relation induced by the assignments in G is acyclic.

Note that any gs-monoidal Σ-space is also a relational Σ-space. As we will see, the terminology 'relational' is due to the application of such spaces to the modeling of relations. For example, in a relational Σ-space each variable can be assigned and used as many times as necessary, i.e., it can be connected to several (possibly to none) other variables.

Proposition 4.7. *The parallel and sequential composition of relational Σ-spaces yield relational Σ-spaces.*

Analogously to the gs-monoidal case, we can characterize coduplicators and codischargers.

Definition 4.6 (*The Σ-spaces $\bar{G}^2_\omega$ and $\bar{G}^0_\omega$*). For each string $\omega \in S^*$, with $|\omega| = n$, we define the relational Σ-spaces $\bar{G}^2_\omega$ and $\bar{G}^0_\omega$ as follows: Let Y and X be lists of names such that $|Y| = 2n$, $|X| = n$, $Y[i] : \omega[i]$, $Y[i+n] : \omega[i]$, and $X[i] : \omega[i]$, for $i = 1, \ldots, n$.

[4] We recall that term graphs [3] are essentially finite directed acyclic graphs labeled over a signature, allowing for the explicit representation of subterm sharing.

Then

$$\bar{G}^0_\omega = \{\Box \mapsto\!\!\!\!\rightarrow \varepsilon,\ X \mapsto\!\!\!\!\rightarrow \Box\},$$

$$\bar{G}^2_\omega = \{\Box \mapsto\!\!\!\!\rightarrow Y,\ X \mapsto\!\!\!\!\rightarrow \Box\} \cup \{Y[i] \mapsto\!\!\!\!\rightarrow X[i] \mid i = 1,\ldots,n\}$$

$$\cup \{Y[i+n] \mapsto\!\!\!\!\rightarrow X[i] \mid i = 1,\ldots,n\}.$$

The families $\{\bar{G}^2_\omega\}_{\omega\in S^*}$ and $\{\bar{G}^0_\omega\}_{\omega\in S^*}$ behave, respectively, as the coduplicator and the codischarger for relational spaces.

Proposition 4.8. *Relational Σ-spaces are the arrows of a concrete r-monoidal category, which is* (*isomorphic in* **RMCat** *to*) *the one freely generated by Σ.*

Proof (*Sketch*). We already know that relational Σ-spaces possess a monoidal category structure. Moreover, the Σ-spaces $G_{\omega,\omega'}$, $\{G^2_\omega\}_{\omega\in S^*}$, and $\{G^0_\omega\}_{\omega\in S^*}$ defined in the previous sections behave as symmetries, duplicators and dischargers also for generic relational spaces. Therefore, we have just to show that the families of Σ-spaces $\{\bar{G}^2_\omega\}_{\omega\in S^*}$ and $\{\bar{G}^0_\omega\}_{\omega\in S^*}$ are transformations, respectively, from $\_1 \otimes \_1$ and from the constant G_ε to the identity functor $\_1$, satisfying the coherence axioms in Definition 2.5. This can be done by exploiting the definition of $\bar{G}^2_\omega$, and $\bar{G}^0_\omega$. To show the freeness of our model we rely on the results of [8], since relational Σ-spaces offer a concrete mathematical structure corresponding to a normalized representation for relations. The proof technique is the standard one: First, a functor is defined by structural induction that goes from the freely generated r-monoidal category to relational Σ-spaces (with the obvious preservation of symmetries, duplicators, dischargers and their 'co' counterparts); second, a normal form is defined for r-monoidal terms and it is shown that whenever two different normal forms are mapped into the same Σ-space, then they can be proved equivalent; eventually, the inductively defined functor is proved surjective. Here we just remark that the normal form for the arrows of the free r-monoidal category associated to Σ has the following shape:

$$r_1; (u_1 \otimes t_1); r_2; (u_2 \otimes t_2); \ldots; r_n; (u_n \otimes t_n); r_{n+1},$$

where:

(1) each r_i is a relation, and therefore finds a normal representation (by the results in [8]) as the term $(\otimes_{j=1}^{n_i} \nabla^{k_{i,j}}_{a_{i,j}}); s_i; (\otimes_{j=1}^{m_i} \Delta^{h_{i,j}}_{b_{i,j}})$, with s_i a permutation, $\nabla^0_a = !_a$, $\Delta^0_a = \dagger_a$, $\nabla^1_a = \Delta^1_a = id_a$, $\nabla^2_a = \nabla_a$, $\Delta^2_a = \Delta_a$, $\nabla^{k+1}_a = \nabla_a; (\nabla^k_a \otimes id_a)$ and $\Delta^{k+1}_a = (\Delta^k_a \otimes id_a); \Delta_a$ if $k \geqslant 2$;
(2) the u_i's are identities;
(3) each t_i is the parallel composition of generators at depth i in the Σ-space associated to the arrow (respecting some predetermined, fixed total ordering of the operators in Σ). □

The additional freedom allowed in the link structure has been used to model *flow-graphs* [8, 39], that is, hyper-graphs labeled over a signature. In fact, since codischargers mimic the occurrence of name matching, many-to-many broadcasting is simulated, as shown by the following result.

Proposition 4.9. *Let Σ be a many-sorted signature, and let $\omega, \omega' \in S^*$ be strings. The class of r-monoidal Σ-spaces with source ω and target ω', containing no generators, is in one-to-one correspondence with the class of (sort-preserving) relations between the components of ω and those of ω'.*

We refer the reader to [8] for more details.

4.4. A case study: finite partial orders

If Σ is a one-sorted signature that contains only unary operators, then we may use relational Σ-spaces for the representation of labeled partial orders.

Definition 4.7. If Σ is a one-sorted signature containing only unary operators, a relational Σ-space G is called a po_Σ-space.

Let $(P, \sqsubseteq, \ell, A)$ be a finite partial order labeled over the set A (hence, $\ell : P \to A$ for P partial order), and let us consider the Σ-spaces over the signature $\Sigma_A = \{a : 1 \to 1 \mid a \in A\}$. In a similar fashion to the proposal of [17], the basic ingredients for the normal form representation are the Σ_A-spaces in the family $\{E_a\}_{a\in A}$, where

$$E_a = \{\Box \mapsto\!\!\!\!\to y_1, x \overset{a}{\mapsto\!\!\!\!\to} y, x_1 \mapsto\!\!\!\!\to \Box, y_1 \mapsto\!\!\!\!\to x, y \mapsto\!\!\!\!\to x_1, y_1 \mapsto\!\!\!\!\to x_1\}.$$

Intuitively, each Σ-space E_a represents an event with label a in parallel with an identity. The presence of the links $y_1 \mapsto\!\!\!\!\to x$ and $y_1 \mapsto\!\!\!\!\to x_1$, acting as a duplicator of the input position, together with the presence of the links $y \mapsto\!\!\!\!\to x_1$ and $y_1 \mapsto\!\!\!\!\to x_1$, acting as a match in the output position, creates a sort of implicit transitive closure of identities whenever the sequential composition is applied. Thus, let spo_{Σ_A}-spaces denote the subclass of po_{Σ_A}-spaces, i.e. those spaces generated by closure of the family $\{E_a\}_{a\in A}$ with respect to the constants and operators of *r-monoidal* spaces. The Σ_A-space associated to P is obtained by composing the E_a's following the intuitive correspondence with the labeled elements of P.

Theorem 4.10. *The class of spo_{Σ_A}-spaces from ε to ε is in one-to-one correspondence with the class of finite partial orders labeled on A.*

Proof (*Sketch*). The mapping from partial orders to Σ-spaces can be defined as follows. If the partial order is made up of disconnected partially ordered subsets, then the associated Σ-space will be the parallel composition of the Σ-spaces associated to the various components (the order of the parallel composition is not important, since they are all arrows from ε to ε and therefore can be freely permuted, i.e., we rely on,

a multi-set structure). If the partial order is connected, then

- we divide the elements in *layers* according to the ordering (minimal elements go in the first layer $\mathscr{L}_1$, the elements whose all predecessors are minimal elements go in the second layer $\mathscr{L}_2$, and so on, in such a way that an element is in the ith layer $\mathscr{L}_i$ iff all its predecessors are in $\bigcup_{j<i}\mathscr{L}_j$ and there is at least one predecessor that is in $\mathscr{L}_{i-1}$) and let m be the total number of layers;
- we fix an arbitrary total order $<$ for the elements in each layer;
- for each layer $\mathscr{L}_i$ with n_i elements ordered as $e_{i,1}<\cdots<e_{i,n_i}$, we define the associated Σ-space H_i as the product $G_{s_i}\otimes E_{a_{i,1}}\otimes\cdots\otimes E_{a_{i,n_i}}$, where $a_{i,j}=\ell(e_{i,j})$ and s_i is the number of elements in $\bigcup_{j<i}\mathscr{L}_j$ (the idea is that also the attach points are propagated through the layers, since this is necessary to model partial orders where an element depends from unrelated elements in different layers);
- we let r_i denote the relation between the elements of the layers $\bigcup_{j\leqslant i}\mathscr{L}_i$ and $\mathscr{L}_{i+1}$ which is obtained by restricting $\sqsubset$, and call R_i the relational Σ-space associated to r_i;
- let $R'_i=(G^2_{s_i+n_i});(G_{s_i+n_i}\otimes R_i)$;
- the Σ-space associated to the connected partial order is then

$$R_0;H_1;R'_1;H_2;R'_2\cdots;R'_{m-1};H_m;R_m,$$

where R_0 is the parallel composition of n_1 copies of the discharger G^0, and R_m is the parallel composition of as many copies of the codischarger $\bar{G}^0$ as the elements in the partial order.

The idea is to have a representative in any H_i (either an identity, or a component E_a) for each element in the first i layers of the partial order, then two consecutive layers i and $i+1$ are connected by relating the elements of the $(i+1)$th layer with the representative of their predecessors, and maintaining the correspondence between the representatives of the same element in the two layers. This definition creates some redundancy (for example when $e_1\sqsubset e_2\sqsubset e_3$ the connection between e_1 and e_3 is added twice: one is expressed directly, and one is inherited from the connection between e_3 and e_2, since $e_1\sqsubset e_2$), which is however discarded in the overall composition (Σ-spaces are sets of sentences and not multi-sets).

The backward mapping is defined by taking an element $e_{(x,a,y)}$ with label a for each generator sentence $x\overset{a}{\mapsto\!\!\!\!\!\rightarrow}y$ in the Σ-space, and by letting

$$e_{(x_1,a_1,y_1)}\sqsubset e_{(x_2,a_2,y_2)}$$

whenever the Σ-space contains a link sentence $y_1\mapsto\!\!\!\!\!\rightarrow x_2$. Since input and output strings are empty, this mapping is injective. □

Example 4.1. The Σ_A-space $\bar{G}^0_1;E_a;G^2_1;(E_b\otimes E_c);\bar{G}^2_1;E_b;G^0_1$ corresponds to the partial order in Fig. 17. The explicit construction for a subterm is given in Fig. 18 (where, for simplicity, we adopt a self-explanatory vector notation for the input and output

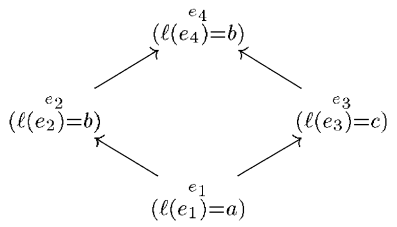


Fig. 17. A partial order $P=\{e_i\}_{i=1,\ldots,4}$ with labels over the set $A=\{a,b,c\}$.

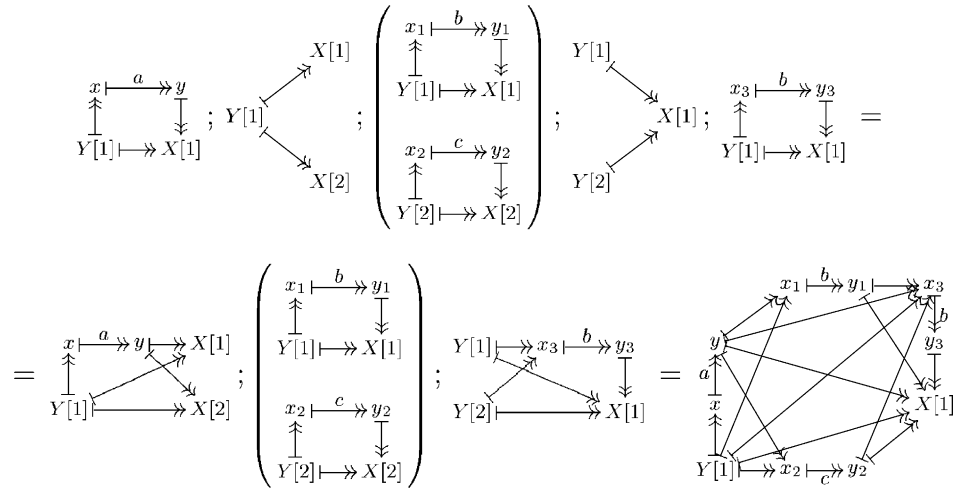


Fig. 18. Step by step illustration of $E_a;G_1^2;(E_b\otimes E_c);\bar{G}_1^2;E_b$.

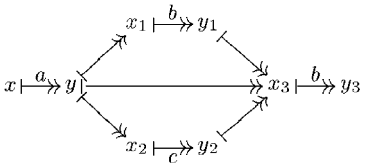


Fig. 19. $\bar{G}_1^0;E_a;G_1^2;(E_b\otimes E_c);\bar{G}_1^2;E_b;G_1^0$.

variables). Notice that this expression is simpler than the one associated to the same partial order by the mapping in the proof of Theorem 4.10, which is

$$\bar{G}_1^0;E_a;(G_1^2;(G_1\otimes G_1^2));(G_1\otimes E_b\otimes E_c);(G_3^2;(G_3\otimes((\bar{G}_1^2\otimes G_1);\bar{G}_1^2));$$

$$(G_3\otimes E_b);(G_1^0\otimes G_1^0\otimes G_1^0\otimes G_1^0),$$

but the two expressions can be proved equivalent, and evaluate to the Σ-space in Fig. 19 (without 'inputs' and 'outputs').

We believe that our characterization of partial orders as 'closed elements' of an axiomatically defined algebra of relations is new. One of the advantages of our representation is given by the straightforward definition of sequential composition for partial orders, which is of course dependent upon the ordering of input and output elements.

4.5. New-bang

The last class of relational spaces that we consider can be used to model processes with creation and deletion.

Definition 4.8 (*New-bang Σ-spaces*). A Σ-space G is called *new-bang* if
(1) all the link sentences have the form $y \mapsto\!\!\!\!\rightarrow x$, for y a result and x a name;
(2) the flow relation induced by the assignments in G is acyclic;
(3) each variable is assigned at most once in G;
(4) each variable is used at most once in G.

Notice that any new-bang Σ-space is also a relational Σ-space. Here we forbid sharing and matching, but allow for the hiding of results and for the introduction of new resources.

Proposition 4.11. *The parallel and sequential composition of new-bang spaces yield new-bang spaces as a result.*

Dischargers G^0_ω and codischargers $\bar{G}^0_\omega$ are defined as for relational spaces.

Proposition 4.12. *New-bang Σ-spaces are the arrows of a concrete new-bang category, which is* (*isomorphic in* **NBCat** *to*) *the one freely generated by Σ.*

As for the case concerning partial orders in Section 4.4, the case of one-sorted signatures containing exclusively unary operators is particularly interesting.

Definition 4.9. If Σ is a one-sorted signature containing only unary operators, a new-bang Σ-space G is called a ms_Σ-space.

Now, let A be a set, and let Σ_A be the one-sorted signature containing the elements of A as unary operators. We can now state a theorem analogous to the correspondence result for partial orders, presented in Theorem 4.10.

Theorem 4.13. *The class of ms_{Σ_A}-spaces from ε to ε is in one-to-one correspondence with the class of finite multisets of strings over A.*

Since the sequential composition of two arrows from ε to ε is equivalent to their parallel composition, and verifies the commutative property, on the homset $[\varepsilon, \varepsilon]$ of new-bang spaces we have the structure of a commutative monoid. Hence, new-bang spaces can be used to model a commutative monoid over a monoidal structure. Notice

that the two units are different (one is the empty space, the other is the empty string), as otherwise the two monoidal structures would collapse.

5. Σ-spaces for partitions and contextual nets

In this section we move away from acyclic spaces. This is achieved by allowing the use of backward links, links between input elements, and links between output elements. However, we still require some kind of 'regularity' over the links. In particular we assume links to define equivalence classes (i.e., they are symmetrically closed, while transitivity is imposed by definition and reflexivity is left implicit).

5.1. Partitions

The last class of spaces we consider gives a framework for partitions.

Definition 5.1. A Σ-space G is called a *partition space* if and only if the reflexive closure of the link relation is an equivalence relation.

To some extent, this requirement corresponds to implicitly consider spaces taken up to equivalence classes of variables.

Proposition 5.1. *The parallel and sequential composition of partition spaces yield partition spaces as a result.*

Due to the peculiar nature of the link relation, symmetries, (co)duplicators, and (co)dischargers still exist, but the constraints of partition spaces force some of them to have a richer structure than the one previously considered. In fact, they can be obtained by freely adding link sentences, in order to obtain the minimal equivalence relation. Overloading the notation, we will use the same symbols to denote such spaces.

For each pair of strings $\omega, \omega' \in S^*$, with $|\omega| = n$ and $|\omega|' = m$, the partition space $G_{\omega,\omega'}$ (which plays the rôle of the symmetry) is explicitly described as follows: Let Y and X be lists of (pairwise disjoint) names such that $|Y| = |X| = n+m$, $Y[i] : (\omega \cdot \omega')[i]$, and $X[i] : (\omega' \cdot \omega)[i]$ for $i = 1, \ldots, n+m$. Then

$$\begin{aligned} G_{\omega,\omega'} = {} & \{\Box \mapsto\!\!\!\!\!\rightarrow Y, X \mapsto\!\!\!\!\!\rightarrow \Box\} \cup \{Y[i] \mapsto\!\!\!\!\!\rightarrow X[i-n] \mid i = n+1, \ldots, n+m\} \\ & \cup \{Y[i] \mapsto\!\!\!\!\!\rightarrow X[i+m] \mid i = 1, \ldots, n\} \cup \{X[i+m] \mapsto\!\!\!\!\!\rightarrow Y[i] \mid i = 1, \ldots, n\} \\ & \cup \{X[i-n] \mapsto\!\!\!\!\!\rightarrow Y[i] \mid i = n+1, \ldots, n+m\}. \end{aligned}$$

For each string $\omega \in S^*$, with $|\omega| = n$, the partition spaces G^2_ω, and G^0_ω are as follows: Let Y and X be lists of names such that $|Y| = n$, $|X| = 2n$, $Y[i] : \omega[i]$, $X[i] : \omega[i]$, and

$X[i+n]:\omega[i]$, for $i=1,\ldots,n$. Then

$$G^0_\omega = \{\Box \mapsto\!\!\!\!\rightarrow Y, \varepsilon \mapsto\!\!\!\!\rightarrow \Box\},$$

$$G^2_\omega = \{\Box \mapsto\!\!\!\!\rightarrow Y, X \mapsto\!\!\!\!\rightarrow \Box\} \cup \{Y[i] \mapsto\!\!\!\!\rightarrow X[i] \mid i=1,\ldots,n\}$$

$$\cup \{Y[i] \mapsto\!\!\!\!\rightarrow X[i+n] \mid i=1,\ldots,n\} \cup \{X[i] \mapsto\!\!\!\!\rightarrow X[i+n] \mid i=1,\ldots,n\}$$

$$\cup \{X[i] \mapsto\!\!\!\!\rightarrow Y[i] \mid i=1,\ldots,n\} \cup \{X[i+n] \mapsto\!\!\!\!\rightarrow Y[i] \mid i=1,\ldots,n\}$$

$$\cup \{X[i+n] \mapsto\!\!\!\!\rightarrow X[i] \mid i=1,\ldots,n\}.$$

And similarly for their 'co' version.

Proposition 5.2. *Partition spaces are the arrows of a concrete p-monoidal category which is* (*isomorphic in* **PMCat** *to*) *the one freely generated from* Σ.

Proof (*Sketch*). The proof is analogous to those given in the previous sections. We just point out the normal form representation for the arrows of the free p-monoidal category generated by Σ:

$$p_1; (u \otimes t); p_2,$$

where:

(1) each p_i is a partition and finds a normal representation as the term $s_i; (\otimes_{j=1}^{n_i} \Delta_{a_{i,j}}^{k_{i,j}}; \nabla_{a_{i,j}}^{h_{i,j}}); s_i'$, with s_i and s_i' permutations, $\nabla_a^0 = !_a$, $\Delta_a^0 = \dagger_a$, $\nabla_a^1 = \Delta_a^1 = id_a$, $\nabla_a^2 = \nabla_a$, $\Delta_a^2 = \Delta_a$, $\nabla_a^{k+1} = \nabla_a; (\nabla_a^k \otimes id_a)$, and $\Delta_a^{k+1} = (\Delta_a^k \otimes id_a); \Delta_a$ if $k \geqslant 2$;
(2) u is the parallel composition of identities (one for each partition class of nodes in the space);
(3) t is the parallel composition of all the generators in the space. □

The normal form presentation for the arrows suggests the correspondence result stated in the proposition below. The idea is that p_1 groups together all the inputs that belong to the same partition and introduces a new node for each class that has no representative in the input list, then as many copies of the classes are made, as it is needed by the left parts of generator sentences, plus one (which is propagated via u, in parallel with t). The term p_2 performs a similar task for outputs.

Proposition 5.3. *Let* Σ *be a many-sorted signature, and let* $\omega, \omega' \in S^*$ *be strings. The class of partition* Σ*-spaces with source* ω *and target* ω', *containing no generators, is in one-to-one correspondence with the class of* (*sort-preserving*) *equivalence classes on the disjoint union of the components of* ω *and those of* ω'.

We believe that our algebraic characterization of partitions is new.

5.2. Contextual nets

As a last case we consider a subclass of partition Σ-spaces.

Definition 5.2. A partition Σ-space G is called *contextual* if
(1) each variable is assigned at least once in G;
(2) each variable is used at least once in G.

Proposition 5.4. *Parallel and sequential composition of contextual Σ-spaces yield contextual Σ-spaces.*

Proposition 5.5. *Contextual Σ-spaces are the arrows of a concrete match-share category, which is (isomorphic in* **MShCat** *to) the one freely generated from Σ.*

The name 'contextual' comes from the use of match-share categories for the representation of *contextual nets*, that is, ordinary Petri nets with *read arcs*, following the paradigm of read/write access to shared resources, where readers are allowed to progress in parallel. The axioms of match-share categories faithfully embed the compositional properties of contextual net processes: The basic idea is that the concurrent access by two transitions to the same place, which is part of the context, is simulated by the creation of two copies of that place, concurrently firing the transitions, and then matching the newly created instances of the place. For more details we refer the reader to [17].

6. Conclusion

In this work we focused on a set-theoretical, intensional description of various algebraic models for the representation of 'concurrent and distributed systems' proposed in the recent literature. Since many of them admit a presentation in terms of enriched symmetric monoidal categories, we looked for a unifying concrete framework based on the same concepts. This led us to a precise characterization of four fundamental aspects of an abstract distributed system: its *input* and *output interfaces*, the *basic functionalities* it provides and the *links* for the connection of subsystems.

As a main result, many of the algebraic approaches found in the literature are precisely characterized in our framework by considering different restrictions on the link sentences. We plan to show how these representation results allow for a uniform translation of these theories into a suitable specification formalism for partial algebras [27], as proved for a class of these structures in [5, 28], the main point being the availability of tools supporting executability.

The idea, already sketched in the Ph.D. Thesis of the first author [4] and in [5] (joint work with José Meseguer), consists of representing Σ-spaces as sets of triples of the form $\langle l_1, op, l_2\rangle$, where l_1 and l_2 are either lists of variables or the symbol $\Box$, and op is either an operator of Σ, or one of the auxiliary constructors **in**, **out**, **link**. The various

restrictions about format and correctness on Σ-spaces can then be translated in such representation. Using e.g. the logical language Maude [9] (of the OBJ family [19]), they can be defined as Boolean predicates, and checked at runtime. Since Maude, as well as other languages supporting algebraic specifications and rewriting mechanisms, allows for efficient rewriting over a mixing of associativity, commutativity, identity and idempotency, it could offer a good basis for exploiting Σ-spaces as a data structure for the modeling of distributed systems, with rewrite rules over them defining the dynamics of the system. The process of specification and prototyping of interesting paradigms could then be accelerated and conveniently tested.

Appendix A. Proofs

In this appendix we give a full proof of Lemma 3.5 and Proposition 4.5. The latter is taken, with minor adjustments, from [11].

A.1. Associativity of merge

We present the proof of Lemma 3.5, which is used in Proposition 3.7 to prove the associativity of the sequential composition of Σ-spaces. For the reader convenience, the statement of the lemma is repeated below:

Lemma A.1. *Let W and W' be two lists of names, and let L_1, L_2 and L_3 be three (transitively closed) sets of links such that names in L_1 and L_2 are disjoint except for names in W, names in L_2 and L_3 are disjoint except for names in W', names in L_1 plus W are disjoint from names in L_3 plus W'. Then,*

$$merge(L_1, W, merge(L_2, W', L_3)) = merge(merge(L_1, W, L_2), W', L_3).$$

Proof. Let $L_1' = merge(L_1, W, L_2)$ and $L_3' = merge(L_2, W', L_3)$. The equality $merge(L_1, W, L_3') = merge(L_1', W', L_3)$ follows by proving separately the two inclusions. We detail the proof for $merge(L_1, W, L_3') \subseteq merge(L_1', W', L_3)$; the converse inclusion can be proved analogously. Let $z_1 \mapsto\!\!\!\!\!\rightarrow z_2 \in merge(L_1, W, L_3')$, we want to show that $z_1 \mapsto\!\!\!\!\!\rightarrow z_2 \in merge(L_1', W', L_3)$. By definition of $merge(L_1, W, L_3')$, $z_1, z_2 \notin W$ and there exists a chain of links

$$z_1 \mapsto\!\!\!\!\!\rightarrow W[i_1] \mapsto\!\!\!\!\!\rightarrow W[i_2] \cdots W[i_n] \mapsto\!\!\!\!\!\rightarrow z_2 \in L_1 \cup L_3'$$

for $n \geqslant 0$. We distinguish three cases:

(1) $n = 0$ and $z_1 \mapsto\!\!\!\!\!\rightarrow z_2 \in L_1$: Since $z_1, z_2 \notin W$, by Lemma 3.4 $z_1 \mapsto\!\!\!\!\!\rightarrow z_2 \in L_1'$. Since names in L_1 and W' are disjoint, we can apply again Lemma 3.4 to conclude that $z_1 \mapsto\!\!\!\!\!\rightarrow z_2 \in merge(L_1', W', L_3)$.

(2) $n = 0$ and $z_1 \mapsto\!\!\!\!\!\rightarrow z_2 \in L_3'$: Therefore $z_1, z_2 \notin W'$ and there exists a chain of links

$$z_1 \mapsto\!\!\!\!\!\rightarrow W'[j_1] \mapsto\!\!\!\!\!\rightarrow W'[j_2] \cdots W'[j_m] \mapsto\!\!\!\!\!\rightarrow z_2 \in L_2 \cup L_3$$

for $m \geqslant 0$. We distinguish three possibilities:

(a) $m = 0$ and $z_1 \mapsto\!\!\!\!\rightarrow z_2 \in L_2$: By Lemma 3.4 it follows that $z_1 \mapsto\!\!\!\!\rightarrow z_2 \in merge(L_1', W', L_3)$ (because $z_1, z_2 \notin W \cup W'$).

(b) $m = 0$ and $z_1 \mapsto\!\!\!\!\rightarrow z_2 \in L_3$: Again, by Lemma 3.4 it follows that $z_1 \mapsto\!\!\!\!\rightarrow z_2 \in merge(L_1', W', L_3)$ (because $z_1, z_2 \notin W \cup W'$).

(c) $m > 0$: Note that if $W'[j_k] \mapsto\!\!\!\!\rightarrow W'[j_{k+1}] \in L_2$, then, by Lemma 3.4, $W'[j_k] \mapsto\!\!\!\!\rightarrow W'[j_{k+1}] \in L_1'$, because names in W' are not in W, thus

$$W'[j_1] \mapsto\!\!\!\!\rightarrow W'[j_2] \cdots W'[j_{m-1}] \mapsto\!\!\!\!\rightarrow W'[j_m] \in L_1' \cup L_3.$$

Analogously, if $z_1 \mapsto\!\!\!\!\rightarrow W'[j_1] \in L_2$, then, by Lemma 3.4, $z_1 \mapsto\!\!\!\!\rightarrow W'[j_1] \in L_1'$, because $z_1, W'[j_1] \notin W$; hence $z_1 \mapsto\!\!\!\!\rightarrow W'[j_1] \in L_1' \cup L_3$. The same reasoning applies to $W'[j_m] \mapsto\!\!\!\!\rightarrow z_2$. Therefore

$$z_1 \mapsto\!\!\!\!\rightarrow W'[j_1] \mapsto\!\!\!\!\rightarrow W'[j_2] \cdots W'[j_m] \mapsto\!\!\!\!\rightarrow z_2 \in L_1' \cup L_3.$$

Since $z_1, z_2 \notin W'$, we can conclude $z_1 \mapsto\!\!\!\!\rightarrow z_2 \in merge(L_1', W', L_3)$.

(3) $n > 0$: This is the most difficult case. We can assume without loss of generality that the links in the chain

$$z_1 \mapsto\!\!\!\!\rightarrow W[i_1] \mapsto\!\!\!\!\rightarrow W[i_2] \cdots W[i_n] \mapsto\!\!\!\!\rightarrow z_2$$

belong alternately to L_1 and L_3', since these sets are closed transitively. Thus, suppose that $z_1 \mapsto\!\!\!\!\rightarrow W[i_1], W[i_n] \mapsto\!\!\!\!\rightarrow z_2 \in L_1$ (the other cases are analogous). Then, $W[i_{2k}] \mapsto\!\!\!\!\rightarrow W[i_{2k+1}] \in L_1$ and $W[i_{2k-1}] \mapsto\!\!\!\!\rightarrow W[i_{2k}] \in L_3'$ for $1 \leqslant k < n/2$. Thus, for each k, either $W[i_{2k-1}] \mapsto\!\!\!\!\rightarrow W[i_{2k}] \in L_2$ or there exists a chain

$$W[i_{2k-1}] \mapsto\!\!\!\!\rightarrow W'[j_{k,1}] \mapsto\!\!\!\!\rightarrow W'[j_{k,2}] \cdots W'[j_{k,m_k}] \mapsto\!\!\!\!\rightarrow W[i_{2k}] \in L_2 \cup L_3$$

with $m_k > 0$ and $W[i_{2k-1}] \mapsto\!\!\!\!\rightarrow W'[j_{k,1}], W'[j_{k,m_k}] \mapsto\!\!\!\!\rightarrow W[i_{2k}] \in L_2$ (because L_3 does not contain names in W). Let $k_1 < k_2 < \cdots < k_l$ be the indices for which the second case holds. If there is no such index ($l = 0$), then

$$z_1 \mapsto\!\!\!\!\rightarrow W[i_1] \mapsto\!\!\!\!\rightarrow W[i_2] \cdots W[i_n] \mapsto\!\!\!\!\rightarrow z_2 \in L_1 \cup L_2$$

and therefore $z_1 \mapsto\!\!\!\!\rightarrow z_2 \in L_1'$ because $z_1, z_2 \notin W$. Then, by Lemma 3.4, $z_1 \mapsto\!\!\!\!\rightarrow z_2 \in merge(L_1', W', L_3)$ because $z_1, z_2 \notin W'$. If $l > 0$, then z_1 and z_2 are connected through a chain of the form

$$z_1 \mapsto\!\!\!\!\rightarrow W[i_1] \mapsto\!\!\!\!\rightarrow W[i_2] \mapsto\!\!\!\!\rightarrow \cdots \mapsto\!\!\!\!\rightarrow W[i_{2k_1-1}] \in L_1 \cup L_2$$

(*) $W[i_{2k_1-1}] \mapsto\!\!\!\!\rightarrow W[j_{k_1,1}] \in L_2$
$W'[j_{k_1,1}] \mapsto\!\!\!\!\rightarrow W'[j_{k_1,2}] \mapsto\!\!\!\!\rightarrow \cdots \mapsto\!\!\!\!\rightarrow W'[j_{k_1,m_{k_1}}] \in L_2 \cup L_3$

(**) $W'[j_{k_1,m_{k_1}}] \mapsto\!\!\!\!\rightarrow W[i_{2k_1}] \in L_2$

$$
\begin{aligned}
&W[i_{2k_1}] \mapsto\!\!\!\!\rightarrow W[i_{2k_1+1}] \mapsto\!\!\!\!\rightarrow W[i_{2k_1+2}] \mapsto\!\!\!\!\rightarrow \cdots \mapsto\!\!\!\!\rightarrow W[i_{2k_2-1}] \in L_1 \cup L_2\\
(*)\quad &W[i_{2k_2-1}] \mapsto\!\!\!\!\rightarrow W'[j_{k_2,1}] \in L_2\\
&W'[j_{k_2,1}] \mapsto\!\!\!\!\rightarrow W'[j_{k_2,2}] \mapsto\!\!\!\!\rightarrow \cdots \mapsto\!\!\!\!\rightarrow W'[j_{k_2,m_{k_2}}] \in L_2 \cup L_3\\
(**)\quad &W'[j_{k_2,m_{k_2}}] \mapsto\!\!\!\!\rightarrow W[i_{2k_2}] \in L_2\\
&W[i_{2k_2}] \mapsto\!\!\!\!\rightarrow W[i_{2k_2+1}] \mapsto\!\!\!\!\rightarrow W[i_{2k_2+2}] \mapsto\!\!\!\!\rightarrow \cdots \mapsto\!\!\!\!\rightarrow W[i_{2k_3-1}] \in L_1 \cup L_2\\
&\vdots\\
&W[i_{2k_{l-1}}] \mapsto\!\!\!\!\rightarrow W[i_{2k_{l-1}+1}] \mapsto\!\!\!\!\rightarrow W[i_{2k_{l-1}+2}] \mapsto\!\!\!\!\rightarrow \cdots \mapsto\!\!\!\!\rightarrow W[i_{2k_l-1}] \in L_1 \cup L_2\\
(*)\quad &W[i_{2k_l-1}] \mapsto\!\!\!\!\rightarrow W'[j_{k_l,1}] \in L_2\\
&W'[j_{k_l,1}] \mapsto\!\!\!\!\rightarrow W'[j_{k_l,2}] \mapsto\!\!\!\!\rightarrow \cdots \mapsto\!\!\!\!\rightarrow W'[j_{k_l,m_{k_l}}] \in L_2 \cup L_3\\
(**)\quad &W'[j_{k_l,m_{k_l}}] \mapsto\!\!\!\!\rightarrow W[i_{2k_l}] \in L_2\\
&W[i_{2k_l}] \mapsto\!\!\!\!\rightarrow W[i_{2k_l+1}] \mapsto\!\!\!\!\rightarrow W[i_{2k_l+2}] \mapsto\!\!\!\!\rightarrow \cdots \mapsto\!\!\!\!\rightarrow W[i_n] \mapsto\!\!\!\!\rightarrow z_2 \in L_1 \cup L_2.
\end{aligned}
$$

Hence we can shift links marked with (*) one line up and links marked with (**) one line down, obtaining

$$
\begin{aligned}
&z_1 \mapsto\!\!\!\!\rightarrow W[i_1] \mapsto\!\!\!\!\rightarrow W[i_2] \mapsto\!\!\!\!\rightarrow \cdots \mapsto\!\!\!\!\rightarrow W[i_{2k_1-1}] \mapsto\!\!\!\!\rightarrow W'[j_{k_1,1}] \in L_1 \cup L_2\\
&W'[j_{k_1,1}] \mapsto\!\!\!\!\rightarrow W'[j_{k_1,2}] \mapsto\!\!\!\!\rightarrow \cdots \mapsto\!\!\!\!\rightarrow W'[j_{k_1,m_{k_1}}] \in L_2 \cup L_3\\
&W'[j_{k_1,m_{k_1}}] \mapsto\!\!\!\!\rightarrow W[i_{2k_1}] \mapsto\!\!\!\!\rightarrow \cdots \mapsto\!\!\!\!\rightarrow W[i_{2k_2-1}] \mapsto\!\!\!\!\rightarrow W'[j_{k_2,1}] \in L_1 \cup L_2\\
&W'[j_{k_2,1}] \mapsto\!\!\!\!\rightarrow W'[j_{k_2,2}] \mapsto\!\!\!\!\rightarrow \cdots \mapsto\!\!\!\!\rightarrow W'[j_{k_2,m_{k_2}}] \in L_2 \cup L_3\\
&W'[j_{k_2,m_{k_2}}] \mapsto\!\!\!\!\rightarrow W[i_{2k_2}] \mapsto\!\!\!\!\rightarrow \cdots \mapsto\!\!\!\!\rightarrow W[i_{2k_3-1}] \mapsto\!\!\!\!\rightarrow W'[j_{k_3,1}] \in L_1 \cup L_2\\
&\vdots\\
&W'[j_{k_{l-1},m_{k_{l-1}}}] \mapsto\!\!\!\!\rightarrow W[i_{2k_{l-1}}] \mapsto\!\!\!\!\rightarrow \cdots \mapsto\!\!\!\!\rightarrow W[i_{2k_l-1}] \mapsto\!\!\!\!\rightarrow W'[j_{k_l,1}] \in L_1 \cup L_2\\
&W'[j_{k_l,1}] \mapsto\!\!\!\!\rightarrow W'[j_{k_l,2}] \mapsto\!\!\!\!\rightarrow \cdots \mapsto\!\!\!\!\rightarrow W'[j_{k_l,m_{k_l}}] \in L_2 \cup L_3\\
&W'[j_{k_l,m_{k_l}}] \mapsto\!\!\!\!\rightarrow W[i_{2k_l}] \mapsto\!\!\!\!\rightarrow W[i_{2k_l+1}] \mapsto\!\!\!\!\rightarrow \cdots \mapsto\!\!\!\!\rightarrow W[i_n] \mapsto\!\!\!\!\rightarrow z_2 \in L_1 \cup L_2.
\end{aligned}
$$

Then, since W' and W are disjoint and $z_1, z_2 \notin W$, we get

$$
\begin{aligned}
&z_1 \mapsto\!\!\!\!\rightarrow W'[j_{k_1,1}] \in L_1'\\
&W'[j_{k_1,m_{k_1}}] \mapsto\!\!\!\!\rightarrow W'[j_{k_2,1}] \in L_1'\\
&W'[j_{k_2,m_{k_2}}] \mapsto\!\!\!\!\rightarrow W'[j_{k_3,1}] \in L_1'\\
&\vdots\\
&W'[j_{k_{l-1},1}] \mapsto\!\!\!\!\rightarrow W'[j_{k_l,1}] \in L_1'\\
&W'[j_{k_l,m_{k_l}}] \mapsto\!\!\!\!\rightarrow z_2 \in L_1'.
\end{aligned}
$$

Moreover, noticing that any link in L_2 between two elements of W' is also in L_1' (by Lemma 3.4 since W and W' are disjoint), we obtain

$$
\begin{aligned}
&W'[j_{k_1,1}] \mapsto\!\!\!\!\rightarrow W'[j_{k_1,2}] \mapsto\!\!\!\!\rightarrow \cdots \mapsto\!\!\!\!\rightarrow W'[j_{k_1,m_{k_1}}] \in L_1' \cup L_3\\
&W'[j_{k_2,1}] \mapsto\!\!\!\!\rightarrow W'[j_{k_2,2}] \mapsto\!\!\!\!\rightarrow \cdots \mapsto\!\!\!\!\rightarrow W'[j_{k_2,m_{k_2}}] \in L_1' \cup L_3\\
&\vdots\\
&W'[j_{k_l,1}] \mapsto\!\!\!\!\rightarrow W'[j_{k_l,2}] \mapsto\!\!\!\!\rightarrow \cdots \mapsto\!\!\!\!\rightarrow W'[j_{k_l,m_{k_l}}] \in L_1' \cup L_3.
\end{aligned}
$$

Therefore we have found a chain in $L_1' \cup L_3$ connecting z_1 and z_2 through the elements of W'. Since $z_1, z_2 \notin W'$, then $z_1 \mapsto\!\!\!\!\rightarrow z_2 \in merge(L_1', W', L_3)$. □

A.2. Building free categories

In this first part we make explicit the construction of both the free symmetric monoidal and the free gs-monoidal category.[5] For the sake of readability, we restrict to the one-sorted hyper-signatures; the many-sorted case can be recovered easily.

Example A.1. The forgetful functor $|\_| : \mathbf{SMCat} \to \mathbf{Set}$ mapping a symmetric monoidal category to its set of objects has a left adjoint SM that maps a set S to the free symmetric monoidal category generated by S. An explicit description of category $SM(S)$ can be given via inference rules and equations. We present them for our specific case of interest, namely $SM(\mathbf{1})$ (where $\mathbf{1}$ is a singleton set).

The category $SM(\mathbf{1}) = \langle \underline{\mathbb{N}}, \otimes, \underline{0}, \gamma \rangle$ has as objects underlined natural numbers $\underline{n} \in \underline{\mathbb{N}}$, on which functor $\otimes$ is defined as $\underline{n} \otimes \underline{m} = \underline{n+m}$ (in fact, objects must form a monoid generated by a singleton set; the unit is clearly $\underline{0}$). Arrows are equivalence classes of terms generated by the following rules:

$$\frac{n \in \mathbb{N}}{id_{\underline{n}} : \underline{n} \to \underline{n}} \qquad \frac{n, m \in \mathbb{N}}{\gamma_{\underline{n},\underline{m}} : \underline{n} \otimes \underline{m} \to \underline{m} \otimes \underline{n}}$$

$$\frac{t : \underline{n} \to \underline{m}, \quad t' : \underline{m} \to \underline{k}}{t; t' : \underline{n} \to \underline{k}} \qquad \frac{t : \underline{n} \to \underline{m}, \quad t' : \underline{n}' \to \underline{m}'}{t \otimes t' : \underline{n} \otimes \underline{n}' \to \underline{m} \otimes \underline{m}'}$$

with respect to the following equations: For all $\underline{n}, \underline{m}, \underline{k}, \underline{n}', \underline{m}' \in \underline{\mathbb{N}}$, and for all arrows $t, t_1, t_2, t_3 \in SM(\mathbf{1})$
(*Categories*):

$$id_{\underline{n}}; t = t = t; id_{\underline{m}} \quad for\ t : \underline{n} \to \underline{m},$$

$$(t; t_1); t_2 = t; (t_1; t_2).$$

(*Functoriality*):

$$id_{\underline{n} \otimes \underline{m}} = id_{\underline{n}} \otimes id_{\underline{m}},$$

$$(t; t_1) \otimes (t_2; t_3) = (t \otimes t_2); (t_1 \otimes t_3) \quad \textit{whenever both sides are defined}.$$

[5] Since we focus on the representation result for Σ-spaces, our presentation is tailored accordingly. The reader will note that our definition of monoidal categories actually characterizes what are called *strict* monoidal categories in the literature. Since no confusion can arise, we have omitted the prefix 'strict' in order to ease the notation. Also, we only consider monoidal functors that preserve monoidal product and unit 'on the nose'.

(*Monoidality*):

$$t \otimes id_{\underline{0}} = t = id_{\underline{0}} \otimes t,$$

$$(t \otimes t_1) \otimes t_2 = t \otimes (t_1 \otimes t_2).$$

(*Naturality*):

$$\gamma_{n',n}; (t \otimes t_1) = (t_1 \otimes t); \gamma_{m',m} \quad for\ t : \underline{n} \to \underline{m}, t_1 : \underline{n}' \to \underline{m}'.$$

(*Symmetry*):

$$\gamma_{\underline{0},\underline{0}} = id_{\underline{0}},$$

$$\gamma_{\underline{n},\underline{m}}; \gamma_{\underline{m},\underline{n}} = id_{\underline{n}\otimes\underline{m}}, \quad and$$

$$\gamma_{\underline{k}\otimes\underline{n},\underline{m}} = (id_{\underline{k}} \otimes \gamma_{\underline{n},\underline{m}}); (\gamma_{\underline{k},\underline{m}} \otimes id_{\underline{n}}).$$

Note also that for all the arrows of $SM(\mathbf{1})$ their source and target actually coincide (that is, for $\underline{n} \neq \underline{m}$, the homset $SM(\mathbf{1})[\underline{n}, \underline{m}]$ is empty).

Intuitively, symmetric monoidal categories formalize in categorical terms the basic notions of *tupling* and *permutation*, as shown by the many coherence results for these categories (originating from [26]). For our purposes, we just need the following result.

Lemma A.2 (Symmetries and permutations). *For any $n \in \mathbb{N}$, the set of arrows from $\underline{n}$ to $\underline{n}$ in $SM(\mathbf{1})$ (i.e., the homset $SM(\mathbf{1})[\underline{n}, \underline{n}]$) is isomorphic to the set of permutations over $\{1, \ldots, n\}$.*

An analogous characterization for the set of *all* possible functions can be obtained, if we enrich the categories at hand.

Example A.2. Similarly to Example A.1, let $GSM : \mathbf{Set} \to \mathbf{GSCat}$ be the left adjoint functor mapping each set to its free gs-monoidal category. An explicit description of $GSM(\mathbf{1}) = \langle \underline{\mathbb{N}}, \otimes, \underline{0}, \gamma, \nabla, ! \rangle$ is easily obtained by adding to the description of $SM(\mathbf{1})$ in Example A.1 the following inference rules for arrows

$$\frac{n \in \mathbb{N}}{!_{\underline{n}} : \underline{n} \to \underline{0}} \qquad \frac{n \in \mathbb{N}}{\nabla_{\underline{n}} : \underline{n} \to \underline{n} \otimes \underline{n}}$$

and the following axioms:[6] For all $\underline{n}, \underline{m}, \underline{k} \in \underline{\mathbb{N}}$,

- $!_{\underline{0}} = \nabla_{\underline{0}} = id_{\underline{0}}$,
- $\nabla_{\underline{n}}; (id_{\underline{n}} \otimes \nabla_{\underline{n}}) = \nabla_{\underline{n}}; (\nabla_{\underline{n}} \otimes id_{\underline{n}})$,
- $\nabla_{\underline{n}}; (id_{\underline{n}} \otimes !_{\underline{n}}) = id_{\underline{n}}$,

[6] Even if these are just instances of the diagrams in Definitions 2.2 and 2.4, we find convenient to summarize them for readers that may prefer an equational presentation for axioms.

- $\nabla_{\underline{n}}; \gamma_{\underline{n},\underline{n}} = \nabla_{\underline{n}}$,
- $\nabla_{\underline{n}\otimes\underline{m}}; (id_{\underline{n}} \otimes \gamma_{\underline{m},\underline{n}} \otimes id_{\underline{m}}) = \nabla_{\underline{n}} \otimes \nabla_{\underline{m}}$,
- $!_{\underline{n}\otimes\underline{m}} = !_{\underline{n}} \otimes !_{\underline{m}}$.

The next result provides an easy characterization for the arrows in $GSM(\mathbf{1})$.

Lemma A.3 (Arrows of $GSM(\mathbf{1})$ are functions). *For any $n, m \in \mathbb{N}$, the set of arrows from $\underline{n}$ to $\underline{m}$ in $GSM(\mathbf{1})$ (i.e., the* homset *$GSM(\mathbf{1})[\underline{n}, \underline{m}]$) is isomorphic to the set of all functions from $\{1,\ldots,m\}$ to $\{1,\ldots,n\}$.*

Proof (*Sketch*). A direct proof of the statement is quite cumbersome but not really needed. It follows from the observation that $GSM(\mathbf{1})$ is a Cartesian category, because the equations $t; \nabla_{\underline{m}} = \nabla_{\underline{n}}; (t \otimes t)$ and $t; !_{\underline{m}} = !_{\underline{n}}$ hold for all $t : \underline{n} \to \underline{m}$ in $GSM(\mathbf{1})$. Hence it is isomorphic to the algebraic theory generated by the empty signature, in which arrows from $\underline{n}$ to $\underline{m}$ are in one-to-one correspondence with m-tuples of variables taken from the set $\{x_1,\ldots,x_n\}$. □

Corollary A.4 (Multiplicators). *For each $n \in \mathbb{N}$, the homset $GSM(\mathbf{1})[\underline{1}, \underline{n}]$ is a singleton: We denote the only arrow in it by ∇^n, and call it a multiplicator. We obviously have $\nabla^0 = !_{\underline{1}}$, $\nabla^1 = id_{\underline{1}}$, and $\nabla^2 = \nabla_{\underline{1}}$.*

Moreover, if $n = m + k + 1$, then $\nabla^n; (id_{\underline{m}} \otimes \nabla_{\underline{1}} \otimes id_{\underline{k}}) = \nabla^{n+1}$ and $\nabla^n; (id_{\underline{m}} \otimes !_{\underline{1}} \otimes id_{\underline{k}}) = \nabla^{n-1}$. Finally, for every permutation $\rho : \underline{n} \to \underline{n}$, we have $\nabla^n; \rho = \nabla^n$.

Let us explain now how a free gs-monoidal category can be generated from a (one-sorted) hyper-signature Σ. The idea is as follows. Every gs-monoidal category M has an underlying *generalized signature* having a monoid of sorts (the objects of M) and all the arrows of M as operators. This defines a forgetful functor from **GSCat** to **GenSig**, the category of generalized signatures. This functor has an obvious left adjoint (denoted **GSTh**) that preserves the monoid of sorts of its argument signature, and generates in a free way all the missing structure on arrows. We will obtain the free gs-monoidal category of a signature Σ as the action of **GSTh** over Σ.

Definition A.1 (*Generalized signatures*). A *generalized signature* Σ is a four-tuple $\langle M, T, s, t\rangle$, where M is a monoid, T a set of operators, and $s, t : T \to U(M)$ are (the source and target) functions, where $U(M)$ is the underlying set of M. A *morphism of generalized signatures* $f : \Sigma \to \Sigma'$ is a pair $\langle f_M : M \to M', f_T : T \to T'\rangle$, where f_M is a monoid homomorphism and f_T a function preserving source and target. Generalized signatures and their morphisms form a *category*, denoted **GenSig**.

Note that a one-sorted signature is a generalized signature where M is the free monoid generated by a singleton (whose elements we denote by underlined natural numbers).

Definition A.2 (*GS-monoidal theories*). Let $V : \mathbf{GSCat} \to \mathbf{GenSig}$ be the forgetful functor mapping a gs-monoidal category to the underlying generalized signature, and let $\mathbf{GSTh} : \mathbf{GenSig} \to \mathbf{GSCat}$ be its left adjoint. Then the *gs-monoidal theory* of a given (one-sorted) hyper-signature Σ is defined as $\mathbf{GSTh}(\Sigma)$, i.e., the free gs-monoidal category generated by Σ.

An explicit description of category $\mathbf{GSTh}(\Sigma)$ for a given signature can be easily provided on the basis of Examples A.1 and A.2. In fact, such a category is obtained by generating arrows using all the inference rules listed in those examples plus the following rule:

$$(\textit{generators}) \quad \frac{f \in \Sigma_{\underline{n},\underline{m}}}{f_\Sigma : \underline{n} \to \underline{m}}$$

and then by imposing all the axioms listed in the examples.

More explicitly, it will be convenient to regard the arrows of $\mathbf{GSTh}(\Sigma)$ as equivalence classes of terms as follows. Let $\Psi(\Sigma)$ be the signature having as constants the (countable) set $\{id_{\underline{n}}, \gamma_{\underline{n},\underline{m}}, \nabla_{\underline{n}}, !_{\underline{n}} \mid n, m \in \mathbb{N}\} \cup \{f_\Sigma \mid f \in \Sigma\}$, and as binary operators the set $\{;, \otimes\}$. Let $\hat{T}_{\Psi(\Sigma)}$ be the set of terms over $\Psi(\Sigma)$ generated by the inference rules of Examples A.1 and A.2, plus the rule (*generators*) above. Then, quite obviously, an arrow t of $GSM(\mathbf{1})$ is an equivalence class of terms $t \in \hat{T}_{\Psi(\Sigma)}$ with respect to the equivalence induced by the axioms of Examples A.1 and A.2.

Note that there is an obvious chain of inclusions $SM(\mathbf{1}) \hookrightarrow GSM(\mathbf{1}) \hookrightarrow \mathbf{GSTh}(\Sigma)$. The following result will be useful in the following.

Lemma A.5 (Decomposition of arrows of gs-monoidal theories). *For any signature Σ, for each arrow t of the associated* gs*-monoidal theory $\mathbf{GSTh}(\Sigma)$, either $t = id_{\underline{0}}$ or it can be decomposed into the composition $t_0; \ldots; t_k$, with $k \geqslant 0$, of arrows of the form $t_i = (\otimes_{j=1}^{n_i} id_{\underline{1}}) \otimes t_i' \otimes (\otimes_{j=1}^{m_i} id_{\underline{1}})$, where $n_i, m_i \geqslant 0$, and $t_i' \in \{id_{\underline{1}}, \gamma_{\underline{1},\underline{1}}, \nabla_{\underline{1}}, !_{\underline{1}}\} \cup \{f_\Sigma \mid f \in \Sigma\}$.*

Proof. One way of proving this is to show that every term in $\hat{T}_{\Psi(\Sigma)}$ is equivalent to a term of the required shape in $\hat{T}_{\Psi_0(\Sigma)}$, where signature $\Psi_0(\Sigma)$ has the (finite) set of constants $\{id_{\underline{1}}, \gamma_{\underline{1},\underline{1}}, \nabla_{\underline{1}}, !_{\underline{1}}\} \cup \{f_\Sigma \mid f \in \Sigma\}$, and $\{;, \otimes\}$ as binary operators.

First, note that all constants in $\{id_{\underline{n}}, \gamma_{\underline{n},\underline{m}}, \nabla_{\underline{n}}, !_{\underline{n}}\}$ are equivalent to terms over $\Psi_0(\Sigma)$. This is obvious for $id_{\underline{n}}$ and $!_{\underline{n}}$; for $\gamma_{\underline{n},\underline{m}}$ and $\nabla_{\underline{n}}$ it can be shown by repeated applications of the last axiom of Example A.1 and of the fifth axiom of Example A.2, exploiting also the fact that $\gamma_{\underline{n},\underline{m}} = \gamma_{\underline{m},\underline{n}}^{-1}$. Since the (binary) operators in $\Psi(\Sigma)$ and $\Psi_0(\Sigma)$ coincide, this is sufficient to show that every term over $\Psi(\Sigma)$ is equivalent to a term over $\Psi_0(\Sigma)$.

Second, any term in $\hat{T}_{\Psi_0(\Sigma)}$ can be transformed into a sequential composition of parallel components by lifting all semicolons to the outermost level, by repeated applications of the second axiom for functoriality in Example A.1; sometimes identities have to be added, as in $t \otimes (t'; t'') = (t; id_{\underline{n}}) \otimes (t'; t'') = (t \otimes t'); ((\otimes_{j=1}^{n} id_{\underline{1}}) \otimes t'')$. Finally the same

technique (i.e., adding identities and applying functoriality) can be used to leave in each component of the sequential decomposition only one basic arrow different from $id_{\underline{1}}$. □

A.3. GS-Monoidal Σ-spaces are arrows of the gs-monoidal theory

In this section we prove the correspondence between the gs-monoidal theory of a signature Σ and the category $\mathbf{GS}_\Sigma$ whose arrows are gs-monoidal Σ-spaces. We restrict again to one-sorted signatures: The many-sorted case can be recovered easily. The first observation is that $\mathbf{GS}_\Sigma$ can be equipped with a gs-monoidal structure, as it is recalled below.

Lemma A.6 (The category of gs-monoidal Σ-spaces). *Let $\mathbf{GS}_\Sigma$ be the category whose objects are (underlined) natural numbers, and arrows are* gs-*monoidal Σ-spaces, such that $G : \underline{n} \to \underline{m}$ if $|in(G)| = n$ and $|out(G)| = m$.*
Then the structure $\langle \mathbf{GS}_\Sigma, \otimes, G_{\underline{0}}, G_{\underline{n},\underline{m}}, G^2_{\underline{n}}, G^0_{\underline{n}} \rangle$ is a gs-*monoidal category.*

Therefore the axioms for gs-monoidality are sound for gs-monoidal Σ-spaces, i.e., any equation that can be deduced from the theory of gs-monoidal categories also holds for gs-monoidal Σ-spaces (via the translation provided by the functor in the next proposition). The result of this appendix implicitly states that those axioms are also complete. For the time being, we can summarize Lemmas A.5 and A.6 with the following proposition.

Proposition A.7 (The full functor for gs-monoidal theories). *For any signature Σ, a full gs-monoidal functor $F_\Sigma : \mathbf{GSTh}(\Sigma) \to \mathbf{GS}_\Sigma$ can be defined inductively.*

Proof. By the freeness of $\mathbf{GSTh}(\Sigma)$ and since $\mathbf{GS}_\Sigma$ is gs-monoidal, a gs-monoidal functor $F_\Sigma : \mathbf{GSTh}(\Sigma) \to \mathbf{GS}_\Sigma$ is uniquely determined by imposing $F_\Sigma(\underline{1}) = \underline{1}$ on objects, and $F_\Sigma(f_\Sigma) = G_f$ on generators (i.e., the arrows corresponding to operators of the signature).

More explicitly, functor F_Σ maps the basic arrows $id_{\underline{0}}$, $id_{\underline{1}}$, $\gamma_{\underline{1},\underline{1}}$, $\nabla_{\underline{1}}$, $!_{\underline{1}}$, and f_Σ to the Σ-spaces G_ε, G_{id}, $G_{\underline{1},\underline{1}}$, $G^2_{\underline{1}}$, $G^0_{\underline{1}}$, and G_f, respectively (for G_f the Σ-space containing only one generator, $x_1, \ldots, x_n \overset{f}{\mapsto\!\!\!\!\!\rightarrow} y_1, \ldots, y_m$ for operator $f \in \Sigma_{\underline{n},\underline{m}}$, input $\square \mapsto\!\!\!\!\!\rightarrow y'_1, \ldots, y'_n$, output $x'_1, \ldots, x'_m \mapsto\!\!\!\!\!\rightarrow \square$ and the set of links $\{y'_i \mapsto\!\!\!\!\!\rightarrow x_i \mid i = 1, \ldots, n\} \cup \{y_i \mapsto\!\!\!\!\!\rightarrow x'_i \mid i = 1, \ldots, m\}$). The functor clearly preserves monoidal product and sequential composition. Lemma A.5 ensures that in this way F_Σ is defined on all arrows of $\mathbf{GSTh}(\Sigma)$. Since any gs-monoidal Σ-space can be decomposed into parallel and sequential composition of elements in its image, it follows that F_Σ is full (i.e., surjective over the homsets). □

Actually, F_Σ is also injective on homsets: in categorical terms, this is expressed by the following proposition.

Lemma A.8 (Faithfulness). *For any signature* Σ, *the functor* $F_\Sigma : \mathbf{GSTh}(\Sigma) \to \mathbf{GS}(\Sigma)$ *is faithful.*

The two propositions imply that the arrows of the homset $\mathbf{GSTh}(\Sigma)[\underline{n}, \underline{m}]$ are in one-to-one correspondence with the gs-monoidal Σ-spaces with m variables in the output, and n variables in the input. And since F_Σ is obviously bijective over objects, our main theorem immediately follows.

Theorem A.9 (Representation theorem for gs-monoidal Σ-spaces). *For any signature* Σ, *the* gs-*monoidal categories* $\mathbf{GSTh}(\Sigma)$ *and* $\mathbf{GS}_\Sigma$ *are isomorphic.*

The rest of this section is devoted to the proof of Lemma A.8. As a first result, it is easy to show that the functor $F_\Sigma : \mathbf{GSTh}(\Sigma) \to \mathbf{GS}_\Sigma$ restricts to an isomorphism between the arrows of $GSM(\mathbf{1})$ and the *discrete* gs-monoidal Σ-spaces, i.e., those Σ-spaces with an empty set of generators.

Proposition A.10 (Faithfulness for the empty signature). *For any signature* Σ, *the functor* $F_\Sigma : \mathbf{GSTh}(\Sigma) \to \mathbf{GS}_\Sigma$ *restricts to a full functor* $F_\emptyset : GSM(\mathbf{1}) \to \mathbf{DGS}_\Sigma$, *where* $\mathbf{DGS}_\Sigma$ *is the subcategory of* $\mathbf{GS}_\Sigma$ *including only discrete* gs-*monoidal* Σ-*spaces as arrows. Moreover*, $F_\emptyset$ *is faithful.*

Proof. An arrow t of $GSM(\mathbf{1}) \subseteq \mathbf{GSTh}(\Sigma)$ is generated by inference rules and equations of Examples A.1 and A.2, thus it does not contain any generator. By the definition of F_Σ in the proof of Proposition A.7, clearly $F_\Sigma(t)$ is discrete, and this ensures that $F_\emptyset : GSM(\mathbf{1}) \to \mathbf{DGS}_\Sigma$ is well defined. Fullness of $F_\emptyset$ follows by instantiating Proposition A.7 for the empty signature $\Sigma_\emptyset$, by observing that $GSM(\mathbf{1}) = \mathbf{GSTh}(\Sigma_\emptyset)$, $\mathbf{DGS}_\Sigma = \mathbf{GS}_{\Sigma_\emptyset}$, and $F_\emptyset = F_{\Sigma_\emptyset}$.

As for faithfulness, we first show that each discrete gs-monoidal Σ-space with a list of n input variables and m output variables identifies a function from $\underline{m}$ to $\underline{n}$. In fact, a discrete space G may only contain links $\{y_i \mapsto\!\!\!\!\to x_j \mid i = 1, \ldots, n,\ j = 1, \ldots, m\}$ for input $\square \mapsto\!\!\!\!\to y_1, \ldots, y_n$ and output $x_1, \ldots, x_m \mapsto\!\!\!\!\to \square$. Thus, we obtain a function $\phi(G) : \underline{m} \to \underline{n}$; it is easy to see that it is well defined, as it does not depend on the concrete representative chosen in G.

Then, faithfulness of $F_\emptyset$ follows because for each $n, m \in \mathbb{N}$ it restricts to a surjective function (by fullness) from $GSM(\mathbf{1})[\underline{n}, \underline{m}]$ to $\mathbf{DGS}_\Sigma[\underline{n}, \underline{m}]$, which are two isomorphic finite sets, by Lemma A.3 and the above discussion. □

Now for the general case, recall that arrows of $\mathbf{GSTh}(\Sigma)$ and $\mathbf{GS}_\Sigma$ are actually equivalence classes. An arrow of $\mathbf{GSTh}(\Sigma)$ is an equivalence class of terms of a suitable signature, as remarked just before Lemma A.5, while a gs-monoidal Σ-space is an isomorphism class of concrete spaces. To prove Lemma A.8 we need to move to a more concrete setting, by considering representatives of those equivalence classes.

The next definition shows how to associate with a concrete representative τ of an arrow t, a concrete gs-monoidal Σ-space $\delta(\tau)$ that is a representative of the gs-monoidal Σ-space $F_\Sigma(t)$, and also a suitable function ξ_τ relating generators in τ to generators of $\delta(\tau)$.

Definition A.3 (*From representatives of arrows to spaces*). Let $\Psi(\Sigma)$ be the signature and $\hat{T}_{\Psi(\Sigma)}$ be the set of terms as defined just before Lemma A.5. It is convenient to regard a term $\tau \in \hat{T}_{\Psi(\Sigma)}$ as a partial function $\tau : \{0,1\}^* \rightharpoonup \Psi(\Sigma)$, such that its domain $\mathcal{O}(\tau)$ (also called the set of *occurrences* of τ) satisfies for $w \in \{0,1\}^*$ and $i \in \{0,1\}$:

(1) $\mathcal{O}(\tau)$ is finite, nonempty, and left-closed (i.e., if $wi \in \mathcal{O}(\tau)$ then $w \in \mathcal{O}(\tau)$);

(2) $wi \in \mathcal{O}(\tau)$ implies $\tau(w) \in \{;, \otimes\}$, and $\tau(w) \in \{;, \otimes\}$ implies $\{w0, w1\} \subseteq \mathcal{O}(\tau)$. [7]

For $\tau \in \hat{T}_{\Psi(\Sigma)}$, let $\mathcal{O}_\Sigma(\tau) \subseteq \{0,1\}^*$ be the set of all *occurrences of generators* in τ, i.e., $\mathcal{O}_\Sigma(\tau) = \{w \in \mathcal{O}(\tau) \mid \tau(w) = f_\Sigma \text{ for some } f \in \Sigma\}$.

Now let t be a fixed arrow in $\mathbf{GSTh}(\Sigma)$. For each term $\tau \in t$ we define a concrete Σ-space $\delta(\tau)$ which is representative of the gs-monoidal Σ-space $F_\Sigma(t)$, and, at the same time, a function $\xi_\tau : \mathcal{O}_\Sigma(\tau) \to gen(\delta(\tau))$, i.e., to the generators of $\delta(\tau)$. Note that for $\delta(\tau)$ we do not need to describe the entire structure, but it is sufficient to show how a concrete Σ-space in $F_\Sigma(t)$ can be determined uniquely. $\delta(\tau)$ and ξ_τ are defined by induction on the structure of τ.

(1) If τ is in $\{id_{\underline{1}}, \gamma_{\underline{1},\underline{1}}, \nabla_{\underline{1}}, !_{\underline{1}}\}$, then let $\delta(\tau)$ be the (only) discrete Σ-space in $F_\Sigma(t)$ having natural numbers as input and output variables, and defined in the intuitive way (for example, for $\nabla_{\underline{1}}$ the input is $\Box \mapsto\!\!\!\!\rightarrow 1$, the output is $2,3 \mapsto\!\!\!\!\rightarrow \Box$, the set of links $\{1 \mapsto\!\!\!\!\rightarrow 2, 1 \mapsto\!\!\!\!\rightarrow 3\}$). Function $\xi_\tau : \mathcal{O}_\Sigma(\tau) \to gen(\delta(\tau))$ is necessarily the empty function, because $\mathcal{O}_\Sigma(\tau) = \emptyset$.

(2) If $\tau = f_\Sigma$ for operator $f \in \Sigma_{\underline{n},\underline{m}}$, then let $\delta(\tau)$ be the (only) in $F_\Sigma(t)$ having $\underline{n+m}$ as set of input and output variables, such that the input is $\Box \mapsto\!\!\!\!\rightarrow 1, \ldots, n$, the output $\Box \mapsto\!\!\!\!\rightarrow n+1, \ldots, n+m$, the generator is $x_1, \ldots, x_n \overset{f}{\mapsto\!\!\!\!\rightarrow} y_1, \ldots, y_m$ and the set of links $\{i \mapsto\!\!\!\!\rightarrow x_i \mid i = 1, \ldots, n\} \cup \{y_i \mapsto\!\!\!\!\rightarrow n+i \mid i = 1, \ldots, m\}$. In this case clearly $\mathcal{O}_\Sigma(\tau) = \{\lambda\}$, where λ is the empty string, and we define $\xi_\tau(\lambda) = x_1, \ldots, x_n \overset{f}{\mapsto\!\!\!\!\rightarrow} y_1, \ldots, y_m$.

(3) If $\tau = \sigma_0 \otimes \sigma_1$, then let $\delta(\tau) = \delta(\sigma_0) \underline{\oplus} \delta(\sigma_1)$, where $\underline{\oplus}$ is almost like the parallel composition of Σ-spaces. Only, to ensure that $\delta(\tau)$ is a well defined Σ-space, we have to provide an explicit construction of the disjoint union. It suffices to provide it for variables, by stating that $var(G_0 \underline{\oplus} G_1) = \{\langle 0, n\rangle \mid n \in var(G_0)\} \cup \{\langle 1, n\rangle \mid n \in var(G_1)\}$.
Next, we clearly have $\mathcal{O}_\Sigma(\tau) = \{0w \mid w \in \mathcal{O}_\Sigma(\sigma_0)\} \cup \{1w \mid w \in \mathcal{O}_\Sigma(\sigma_1)\}$, and we define ξ_τ as $\xi_\tau(0w) = \langle 0, \xi_{\sigma_0}(w)\rangle$, and $\xi_\tau(1w) = \langle 1, \xi_{\sigma_1}(w)\rangle$.

(4) If $\tau = \sigma_0; \sigma_1$, then we can follow the same idea as in the previous case to define $\delta(\tau)$ and ξ_τ. More precisely, $\delta(\tau)$ can be defined uniquely by making the operation of sequential composition of Σ-spaces deterministic, for example using an explicit construction of disjoint union as in the previous case; and ξ_τ will then be uniquely determined by ξ_{σ_0} and ξ_{σ_1}.

[7] Informally, $\mathcal{O}(\tau)$ is the set of nodes of the syntactic tree of τ, represented by their access paths.

By construction, function ξ_τ is easily shown to be an isomorphism between $\mathcal{O}_\Sigma(\tau)$ and the set of generators of $\delta(\tau)$.

The next lemma shows that for each arrow t of $\mathbf{GSTh}(\Sigma)$, we can choose a term $\tau \in t$ having a suitable structure that mirrors, to some extent, the structure of the gs-monoidal Σ-space associated with t. For the rest of the section, $\rho, \rho', \rho_i, \ldots$ range over permutations, i.e., arrows of $SM(\mathbf{1})$.

Lemma A.11 (Decomposition lemma). *Let Σ be a signature and $t : \underline{n} \to \underline{m}$ be an arrow of the associated* gs-*monoidal theory* $\mathbf{GSTh}(\Sigma)$, *different from* $id_{\underline{0}}$.

(1) *There exists a natural number $q \geqslant 0$ and a term $\tau \in t$ over $\Psi(\Sigma)$, such that $\tau = \tau_0; \tau_1; \ldots; \tau_q$,*[8] *where* $\tau_0 = ((\otimes_{j=1}^{n} \nabla^{k_{0,j}}); \rho_0)$, *and for each* $i \in \underline{q}$, $\tau_i = (id_{\underline{p_i}} \otimes (\otimes_{j=1}^{l_i} (f_\Sigma^{i,j}; \nabla^{k_{i,j}}))); \rho_i$, *for suitable natural numbers p_i, $k_{i,j} \geqslant 0$ and $l_i \geqslant 1$, and function symbols $f_\Sigma^{i,j}$ in Σ. We call $\nabla^{k_{i,j}}$ the multiplicator of $f_\Sigma^{i,j}$, ρ_i* the permutation of level i, *and q the* depth of τ.
For each $w \in \mathcal{O}_\Sigma(\tau)$ (i.e., an occurrence of an operator of Σ in τ), we say that the level *of w in the decomposition is i if and only if w is the occurrence of an operator appearing in τ_i.*[9]

(2) *For a term τ as in the previous point, and $w, v \in \mathcal{O}_\Sigma(\tau)$, let $w <_\tau v$ iff the level of w is lower than the level of v. Moreover, let $w \sqsubset_\tau v$ iff there is a nonempty path in $\delta(\tau)$ from generator $\xi_\tau(v)$ to generator $\xi_\tau(w)$ (see Definition* A.3 *for the notation).*
Then for each $w, v \in \mathcal{O}_\Sigma(\tau)$, $w \sqsubset_\tau v$ implies $w <_\tau v$. In other words, the existence of a path between two generators of a Σ-space implies that in any term representing it, and having the form described in (1), *the* (*operator corresponding to the*) *source generator appears in a higher level than the target generator.*

(3) *Let the* space level *of an occurrence $w \in \mathcal{O}_\Sigma(\tau)$ be the length of the longest path starting from $\xi_\tau(w)$ and ending in a generator, plus one. Moreover, let the* space depth *of τ be the maximal space level for the occurrences in $\mathcal{O}_\Sigma(\tau)$. Then for each τ with a structure as in point* (1) *above, it is possible to build an equivalent term σ having a similar structure, such that* (i) *the depth of σ is equal to the space depth of τ, and* (ii) *for each $w \in \mathcal{O}_\Sigma(\sigma)$ the level of w is equal to its space level.*

Proof. Let $t : \underline{n} \to \underline{m}$ be an arrow of $\mathbf{GSTh}(\Sigma)$ different from id_0.

(1) By Lemma A.5 t includes a term σ on $\Psi_0(\Sigma)$ of the form $\sigma = \sigma_0; \sigma_1; \ldots; \sigma_k$ with $k \geqslant 1$, such that for all $i \leqslant k$, $\sigma_i = (id_{\underline{n_i}} \otimes \sigma_i' \otimes id_{\underline{m_i}})$, where $n_i, m_i \geqslant 0$ and $\sigma_i' \in \{id_{\underline{1}}, \gamma_{\underline{1},\underline{1}}, \nabla_{\underline{1}}, !_{\underline{1}}\} \cup \{f_\Sigma \mid f \in \Sigma\}$. We show by induction on k how σ can be

[8] When $\otimes$ and ; denote binary operators of $\Psi(\Sigma)$, formally speaking they are not associative. We simplify the notation by assuming that they associate to the right. Thus, for example, a term $\tau_0; \tau_1; \tau_2$ over $\Psi(\Sigma)$ should be read as $(\tau_0; (\tau_1; \tau_2))$.

[9] More formally, the level of $w \in \mathcal{O}_\Sigma(\tau) \subseteq \{0,1\}^*$ is the greatest $i \in \underline{q}$ such that 1^i is a prefix of w.

transformed into an equivalent term $\tau = \tau_0; \ldots; \tau_q$ over $\Psi(\Sigma)$ having the required structure.

Base case: Let $k = 0$, thus $\sigma = \sigma_0 = (id_{\underline{n_0}} \otimes \sigma'_0 \otimes id_{\underline{m_0}})$. If $\sigma'_0 \in \{id_{\underline{1}}, \gamma_{\underline{1},\underline{1}}, \nabla_{\underline{1}}, !_{\underline{1}}\}$, then a term τ_0 equivalent to σ and having the desired structure is easily obtained, recalling that $!_{\underline{1}} = \nabla^0$, $id_{\underline{1}} = \nabla^1$, and $\nabla_{\underline{1}} = \nabla^2$. If $\sigma'_0 = f_\Sigma$ and $f \in \Sigma_{\underline{n},\underline{m}}$, then it is readily checked that σ is equivalent to term $\tau_0; \tau_1$, with $\tau_0 = id_{\underline{n_0}} \otimes \gamma_{\underline{n},\underline{m_0}}$ and $\tau_1 = (id_{\underline{n_0+m_0}} \otimes f_\Sigma); (id_{\underline{n_0}} \otimes \gamma_{\underline{m_0},\underline{m}})$, having the required structure.

Inductive case: Let $\sigma = \sigma_0; \ldots; \sigma_k$ for $k > 0$. By induction hypothesis we can assume that $\sigma = \tau_0; \ldots; \tau_q; \sigma_k$, where $\sigma_k = (id_{\underline{n_k}} \otimes \sigma'_k \otimes id_{\underline{m_k}})$, and $\tau_0; \ldots; \tau_q$ has the structure described in the statement.
We proceed by case analysis on σ'_k. If $\sigma'_k \in \{id_{\underline{1}}, \gamma_{\underline{1},\underline{1}}\}$, then σ_k is a permutation and we are done because it can be merged with the permutation ρ_q of τ_q, providing the required decomposition. If $\sigma'_k = f_\Sigma$, then as in the *Base case* σ_k can be decomposed as the sequential composition of two terms, say $\tau'_q; \tau_{q+1}$. Since τ'_q is actually a permutation and τ_{q+1} has the desired structure, we get the required decomposition of σ as $\tau_0; \ldots; (\tau_q; \tau'_q); \tau_{q+1}$, where we consider τ'_q as merged with permutation ρ_q. Finally, suppose that $\sigma'_k \in \{\nabla_{\underline{1}}, !_{\underline{1}}\}$. By naturality of symmetries and identities, σ_k can be shifted towards left along the decomposition $\tau_0; \ldots; \tau_q$ till when σ'_k happens to be composed with a multiplicator, by which it will be 'absorbed' as described in Corollary A.4.

(2) It is easy to check that in the sequential composition $G; G'$ of two gs-monoidal Σ-spaces there cannot be any path from a generator of G to a generator of G', because of acyclicity. The statement follows because the inductive construction of $\delta(\tau)$ interprets the operator ';' on terms as (a concrete version of) composition on concrete Σ-spaces.

(3) Let τ be a term with the structure as in point (1) above. From point (2) we deduce that the depth of τ is greater than or equal to the space depth of τ, and that for each $w \in \mathcal{O}_\Sigma(\tau)$ the level of w is greater than or equal to its space level. Clearly, if for each $w \in \mathcal{O}_\Sigma(\tau)$ its level and space level are equal, then the depth and the space depth of τ are equal and we are done.
Now let w be an occurrence of a generator of τ of minimal level (say i) such that its level differs from its space level (that can be at most $i-1$). Informally, the operator occurrence $\tau(w)$ appearing in τ_i has all its argument operators at most in τ_{i-2} (because by hypothesis their levels and space levels are equal), which implies that operator $\tau(w)$ composes only with identities of level $i-1$. Thus it is possible to shift $\tau(w)$ (and the associated multiplicator) to level $i-1$ by readjusting the identities of levels $i-1$ and i, as well as all involved permutations; in particular, if $\tau(w)$ was the only operator in τ_i, to obtain the desired structure it is necessary to merge the new level i (which is equivalent to a permutation) with the permutation of level $i-1$, and the resulting term will have a depth smaller than τ (in fact, note that in the structure of terms considered in point 1, there must be at least

one operator per level). This process can be iterated and eventually the desired equivalent term σ is obtained; the termination of this algorithm follows by the observation that the quantity $\Sigma_{w \in \mathcal{O}_\Sigma(\tau)} level(w) - space\_level(w)$ (straightforwardly defined) decreases at each iteration, and it must be positive. □

We have now all the ingredients needed to prove Lemma A.8.

Proof of the Faithfulness Lemma (Lemma A.8). Let s and t be two arrows of **GSTh**(Σ), such that $F_\Sigma(s) = F_\Sigma(t)$. We have to show that $s = t$, or, equivalently, that there exist two terms over $\Psi(\Sigma)$, $\sigma \in s$ and $\tau \in t$, which can be proved equivalent using the axioms of Examples A.1 and A.2.

By Lemma A.11, we can assume that both $\sigma \in s$ and $\tau \in t$ have the structure described in point (3). Let $\delta(\sigma)$ and $\delta(\tau)$ be the Σ-spaces associated with σ and τ, respectively, by Definition A.3. By hypothesis they belong to the same Σ-space, thus there exists an isomorphism $\eta : \delta(\sigma) \to \delta(\tau)$. By composing it with the functions ξ_σ and ξ_τ of Definition A.3, we get an isomorphism $\chi \stackrel{\text{def}}{=} \xi_\tau^{-1} \circ \eta \circ \xi_\sigma : \mathcal{O}_\Sigma(\sigma) \to \mathcal{O}_\Sigma(\tau)$ relating occurrences of generators in σ and in τ.

By the choice of σ and τ, since by hypothesis they are mapped to the same Σ-space, we know that they have the same depth and the same number of generators in the corresponding levels. Furthermore, by inspection one may verify that σ and τ have exactly the same set of occurrences of generators. As a consequence the isomorphism χ can be split into a family of isomorphisms $\{\chi_i\}_{i \leqslant l}$ where l is the depth, and χ_i relates generators of level i in σ and τ. Now it is not difficult to transform such isomorphisms into identities, by transforming for example τ into an equivalent term τ' having exactly the same structure of τ. The idea is that two adjacent generators (at once with the corresponding multiplicators) having the same level i can be exchanged by modifying only the permutations of levels $i-1$ and i.

At this point we are left with terms σ and τ' having the same set of occurrences of operators and the same generators in the corresponding occurrences. By inspecting the structure and the relationship with the associated equivalent concrete Σ-spaces, it is easy to see that also all multiplicators in the corresponding occurrences have to be identical, because each of them is determined by the degree of sharing of a specific variable. So the only thing that can differ in terms σ and τ' are the permutations of the various levels. Let us show, by induction on the depth, how τ' can be transformed into σ via suitable manipulations that preserve the equivalence: This concludes the proof.

If the depth of σ and τ' is 0, then we are done by Proposition A.10 because they are discrete. Now assume that their depth is $l+1$. Let Π_σ and $\Pi_{\tau'}$ be the permutations corresponding to the permutations of level $l+1$ of σ and τ', respectively. If they are equal, then σ and τ' are identical at level $l+1$, and we are done because we can make identical their subterms of depth l by induction hypothesis. Now suppose that x is the least argument on which Π_σ^{-1} and $\Pi_{\tau'}^{-1}$ differ. x represents a variable in the output of the corresponding Σ-space. Now either its generator is of level $l+1$ or it is of a lower level. In the first case, we could show that by composing τ' with a permutation

that exchanges $\Pi_\sigma^{-1}(x)$ and $\Pi_{\tau'}^{-1}(x)$, we obtain an arrow τ'' equivalent to τ' such that Π_σ^{-1} and $\Pi_{\tau''}^{-1}$ agree on all $y \leqslant x$ (because both $\Pi_\sigma^{-1}(x)$ and $\Pi_{\tau'}^{-1}(x)$ must be in the target of the same multiplicator of level $l+1$, and by applying the last equation of Corollary A.4). In the second case, if x is a variable in the output corresponding to a generator in a level lower than $l+1$, then both $\Pi_\sigma^{-1}(x)$ and $\Pi_{\tau'}^{-1}(x)$ must be the target of identities of level $l+1$. In this case we can exchange them by inserting a permutation at level $l+1$, and the inverse permutation at level l. The identities of level $l+1$ are clearly not affected, and we have obtained an arrow satisfying the same properties as τ'' above.

By iterating this procedure we finally get a term having the same permutation of level $l+1$ as σ, and the statement follows by induction hypothesis. □

References

[1] S. Abramsky, S. Gay, R. Nagarajan, Interaction categories and the foundations of typed concurrent programming, in: M. Broy (Ed.), Proc. 1994 Marktoberdorf Summer School on Deductive Program Design, Nato ASI Series F, Vol. 152, Springer, Berlin, 1996, pp. 403–442.

[2] Z.M. Ariola, J.W. Klop, Equational term graph rewriting, Fundamenta Informaticae 26 (1996) 207–240.

[3] H.P. Barendregt, M.C.J.D. van Eekelen, J.R.W. Glauert, J.R. Kennaway, M.J. Plasmeijer, M.R. Sleep, Term graph rewriting, in: J.W. de Bakker, A.J. Nijman, P.C. Treleaven (Eds.), Proc. PARLE'87, Parallel Architectures and Languages Europe, Lecture Notes in Computer Science, Vol. 259, Springer, Berlin, 1987, pp. 141–158.

[4] R. Bruni, Tile Logic for Synchronized Rewriting of Concurrent Systems, Ph.D. Thesis TD-01/99, University of Pisa, Department of Computer Science, 1998.

[5] R. Bruni, J. Meseguer, U. Montanari, Process and term tile logic, Technical Report SRI-CSL-98-06, SRI International, 1998, Also Technical Report TR-98-09, University of Pisa, Department of Computer Science, 1998.

[6] R. Bruni, U. Montanari, J. Meseguer, V. Sassone, Functorial semantics of Petri nets under the individual token philosophy, in: M. Hofmann, D. Pavlović, G. Rosolini (Eds.), Proc. CTCS'99, Category Theory and Computer Science, Electronic Notes in Theoretical Computer Science, Vol. 29, Elsevier Sciences, Amsterdam, 1999.

[7] V.-E. Căzănescu, Gh. Ştefănescu, Towards a new algebraic foundation of flowchart scheme theory, Fundamenta Informaticae 13 (1990) 171–210.

[8] V.-E. Căzănescu, Gh. Ştefănescu, Classes of finite relations as initial abstract data types I, Discrete Mathematics 90 (1991) 233–265.

[9] M. Clavel, F. Durán, S. Eker, P. Lincoln, N. Martí-Oliet, J. Meseguer, J. Quesada, Maude: Specification and programming in rewriting logic, available at `http://maude.csl.sri.com/manual`, 1999.

[10] A. Corradini, F. Gadducci, A 2-categorical presentation of term graph rewriting, in: E. Moggi, G. Rosolini (Eds.), Proc. CTCS'97, Category Theory and Computer Science, Lecture Notes in Computer Science, Vol. 1290, Springer, Berlin, 1997, pp. 87–105.

[11] A. Corradini, F. Gadducci, An algebraic presentation of term graphs, via gs-monoidal categories, Appl. Categor. Struct. 7 (1999) 299–331.

[12] A. Corradini, F. Gadducci, Functorial semantics for multi-algebras, in: J.L. Fiadeiro (Ed.), Proc. WADT'98, Recent Trends in Algebraic Development Techniques, Lecture Notes in Computer Science, Vol. 1589, Springer, Berlin, 1999, pp. 78–90.

[13] P. Degano, J. Meseguer, U. Montanari, Axiomatizing the algebra of net computations and processes, Acta Informatica 33 (1996) 641–667.

[14] G. Ferrari, U. Montanari, Tile formats for located and mobile systems, Inform. Comput. 156 (2000) 173–235.

[15] F. Gadducci, R. Heckel, An inductive view of graph transformation, in: F. Parisi-Presicce (Ed.), Proc. WADT'97, Recent Trends in Algebraic Development Techniques, Lecture Notes in Computer Science, Vol. 1376, Springer, Berlin, 1998, pp. 219–233.

[16] F. Gadducci, R. Heckel, M. Llabrés, A bi-categorical axiomatisation of concurrent graph rewriting, in: M. Hofmann, D. Pavlovič, G. Rosolini (Eds.), Proc. CTCS'99, Category Theory and Computer Science, Electronic Notes in Theoretical Computer Science, Vol. 29, Elsevier Sciences, Amsterdam, 1999.
[17] F. Gadducci, U. Montanari, Axioms for contextual net processes, in: K.G. Larsen, S. Skyum, G. Winskel (Eds.), Proc. ICALP'98, Automata, Languages and Programming, Lecture Notes in Computer Science, Vol. 1443, Springer, Berlin, 1998, pp. 296–308.
[18] F. Gadducci, U. Montanari, The tile model, in: G. Plotkin, C. Stirling, M. Tofte (Eds.), Proof, Language and Interaction: Essays in Honour of Robin Milner, MIT Press, Cambridge, MA, 2000.
[19] J. Goguen, G. Malcolm, Algebraic Semantics of Imperative Programs, MIT Press, Cambridge, MA, 1997.
[20] M. Hasegawa, Recursion from cyclic sharing: Traced monoidal categories and models of cyclic lambda-calculus, in: Ph. de Groote, R. Hindly (Eds.), Proc. TLCA'97, Typed Lambda Calculi and Applications, Lecture Notes in Computer Science, Vol. 1210, Springer, Berlin, 1997, pp. 196–213.
[21] H.-J. Hoenke, On partial algebras, in: B. Csákány, E. Fried, E.T. Schmidt (Eds.), Universal Algebra, Colloquia Mathematica Societatis János Bolyai, Vol. 29, North-Holland, Amsterdam, 1977, pp. 373–412.
[22] P. Katis, N. Sabadini, R.F.C. Walters, Bicategories of processes, J. Pure Appl. Algeb. 115 (1997) 141–178.
[23] P. Katis, N. Sabadini, R.F.C. Walters, SPAN(Graph): A categorical algebra of transition systems, in: M. Johnson (Ed.), Proc. AMAST'97, Algebraic Methodology and Software Technology, Lecture Notes in Computer Science, Vol. 1349, Springer, Berlin, 1997, pp. 307–321.
[24] F.W. Lawvere, Functorial semantics of algebraic theories, Proc. Nat. Acad. Sci. 50 (1963) 869–872.
[25] S. Mac Lane, Categories for the Working Mathematician, Springer, Berlin, 1971.
[26] S. MacLane, Natural associativity and commutativity, Rice Univ. Stud. 49 (1963) 28–46.
[27] J. Meseguer, Membership algebra as a logical framework for equational specification, in: F. Parisi-Presicce (Ed.), Proc. WADT'97, Recent Trends in Algebraic Development Techniques, Lecture Notes in Computer Science, Vol. 1376, Springer, Berlin, 1998, pp. 18–61.
[28] J. Meseguer, U. Montanari, Mapping tile logic into rewriting logic, in: F. Parisi-Presicce (Ed.), Proc. WADT'97, Recent Trends in Algebraic Development Techniques, Lecture Notes in Computer Science, Vol. 1376, Springer, Berlin, 1998, pp. 219–233.
[29] J. Meseguer, U. Montanari, V. Sassone, Representation theorems for Petri nets, in: C. Freksa, M. Jantzen, R. Valk (Eds.), Foundations of Computer Science: Potential–Theory–Cognition, to Wilfried Brauer on the occasion of his sixtieth birthday, Lecture Notes in Computer Science, Vol. 1337, Springer, Berlin, 1997, pp. 239–249.
[30] R. Milner, Calculi for interaction, Acta Informatica 33 (1996) 707–737.
[31] U. Montanari, F. Rossi, Contextual nets, Acta Informatica 32 (1995) 545–596.
[32] U. Montanari, F. Rossi, Graph rewriting, constraint solving and tiles for coordinating distributed systems, Appl. Categor. Struct. 7 (1999) 333–370.
[33] M. Pfender, Universal algebra in s-monoidal categories, Technical Report 95-22, University of Munich, Department of Mathematics, 1974.
[34] J. Power, E. Robinson, Premonoidal categories and notions of computation, Math. Struct. Comput. Sci. 7 (1998) 453–468.
[35] E. Robinson, G. Rosolini, Categories of partial maps, Inform. Comput. 79 (1988) 95–130.
[36] V. Sassone, On the Semantics of Petri Nets: Processes, Unfolding and Infinite Computations. Ph.D. Thesis TD-06/94, University of Pisa, Department of Computer Science, 1994.
[37] V. Sassone, An axiomatization of the algebra of Petri net concatenable processes, Theoret. Comput. Sci. 170 (1996) 277–296.
[38] Gh. Ştefănescu, On flowchart theories: Part II. The nondeterministic case, Theoret. Comput. Sci. 52 (1987) 307–340.
[39] Gh. Ştefănescu, Network Algebra, Springer, Berlin, 2000.